

[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

Home

[Edit](#)[New page](#)[Jump to bottom](#)

Nina Karavanić edited this page 3 hours ago · [3 revisions](#)

Programsko inženjerstvo ak.god 2024./2025

Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

PlanBot

Tim: <TG11.2>

- Nina Karavanić
- Josip Galić
- Lea Gojšić
- Ema Mamić
- Borna Maričak
- Toma Begović

PlanBot

Nastavnik: Vlado Sruk

+ Add a custom footer

[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

1. Opis projektnog zadatka

[Edit](#)[New page](#)[Jump to bottom](#)

JosipGalic edited this page 2 days ago · [11 revisions](#)

Cilj projekta je razviti aplikaciju koja korisnicima omogućava jednostavno i učinkovito organiziranje grupnih događaja, bilo da se radi o online sastancima ili događajima uživo. Glavna svrha aplikacije je olakšati korisnicima proces dogovaranja kroz centraliziranu platformu koja integrira mogućnosti predlaganja, glasanja i finaliziranja termina i lokacije događanja. Aplikacija pruža opcije registracije i prijave korisnika, upravljanja događajima, te fleksibilne funkcionalnosti za prilagodbu rasporeda korisnicima. Jedna od ključnih funkcionalnosti aplikacije je integracija s osobnim kalendarima korisnika, poput Google Kalendar-a, čime se omogućava sinkronizacija svih aktivnosti. Ova integracija omogućuje korisnicima jednostavan pregled svih događaja na jednom mjestu i olakšava planiranje kroz bolje razumijevanje raspoloživih termina svih uključenih sudionika.

Problem u organizaciji grupnih događaja leži u složenosti koordinacije između sudionika koji često imaju različite obveze i raspoloživost. Tradicionalne metode organizacije događaja, poput slanja e-pošte ili poruka u grupnim chatovima, često su neučinkovite. Takav način dogovaranja rezultira dugotrajnim razmjenama poruka, nejasnim dogovorima i mogućim preklapanjem obveza sudionika. Izostanak centraliziranog sustava dodatno otežava usklađivanje termina, zbog čega se proces dogovaranja može produžiti. Ovaj projekt adresira te izazove kroz centraliziranu platformu koja nudi transparentnu koordinaciju i učinkovitiji način organizacije.

Jedna od glavnih prednosti ove aplikacije je smanjenje vremena i truda potrebnog za organizaciju događaja. Platforma omogućuje jednostavan pregled slobodnih termina kroz kalendarsku integraciju, dok sustav obavijesti pravovremeno obavještava korisnike o novim događajima ili promjenama u postojećima. Korisnici dobivaju podsjetnike putem e-maila o svim novim objavama i budućim događajima, čime se povećava pravovremenost i smanjuje vjerljivost propuštanja važnih obavijesti. Transparentnost koju nudi aplikacija smanjuje nesporazume među korisnicima, a opcije komentiranja i predlaganja novih termina dodatno olakšavaju suradnju. Korisnici tako mogu aktivno sudjelovati u planiranju i davati svoje prijedloge za lokacije i vremena, čime se osigurava da svi sudionici imaju priliku iznijeti svoje preferencije.

Postoje različita rješenja koja već djelomično omogućuju organizaciju grupnih događaja, no svako od njih ima specifična ograničenja koja ih razlikuju od predložene aplikacije. Jedan od najpoznatijih alata za zakazivanje termina je Doodle, koji omogućava korisnicima da predlože različite termine i glasaju za onaj koji im najviše odgovara. Iako Doodle olakšava dogovaranje termina, nema napredne funkcionalnosti kao što su integracija s kalendarom, predlaganje lokacija ili dodavanje komentara, što su ključne značajke predložene aplikacije. S druge strane, platforma poput Eventbrite-a namijenjena je organizaciji i promociji većih događanja, često komercijalnih prirode, gdje organizatori mogu prodavati ulaznice i pratiti broj prijava. Međutim, fokus Eventbrite-a je na javnim događajima, a ne na privatnim grupnim okupljanjima i ne nudi prilagodljive opcije kao što su predlaganje termina ili personalizacija prema sudionicima. Google Calendar je još jedno popularno rješenje koje omogućuje stvaranje i dijeljenje događaja te integraciju s drugim aplikacijama. Iako je koristan za organizaciju događanja, Google Calendar

nema funkcionalnosti poput glasanja, predlaganja alternativnih termina ili dodavanja komentara na način na koji to omogućuje predložena aplikacija. Meetup, također popularna platforma, namijenjena je pronalasku događaja prema interesima i često okuplja korisnike u grupe na temelju zajedničkih interesa. Dok korisnici mogu označiti prisustvovanje i komentirati događaje, Meetup se više fokusira na stvaranje zajednica i pronalazak događaja nego na fleksibilno planiranje privatnih okupljanja. Poslovne platforme poput Microsoft Teamsa i Zooma pružaju osnovne mogućnosti zakazivanja i integraciju s kalendarom, no te platforme su više usmjerene na poslovnu komunikaciju i ne nude glasanje o terminima ili opcije za predlaganje novih termina i lokacija, što su funkcionalnosti koje predložena aplikacija nudi za bolju prilagodbu i suradnju među korisnicima. Slično tome, Facebook Events omogućava korisnicima kreiranje događaja, dijeljenje informacija i pozivanje gostiju, no nije namijenjen za detaljno planiranje privatnih sastanaka i ne uključuje opcije poput glasanja za termine ili predlaganja alternativnih lokacija. Zaključno, iako postoje različita rješenja koja dijelom pokrivaju neke funkcionalnosti predložene aplikacije, ona često imaju specifična ograničenja. Predložena aplikacija kombinira ključne funkcionalnosti – glasanje o terminima i lokacijama, integraciju s kalendarima, personalizaciju događaja, komentare i dodatne prijedloge – čime korisnicima omogućava potpunu kontrolu nad organizacijom privatnih događaja. Tako predložena aplikacija postaje intuitivan i fleksibilan alat koji nadilazi mogućnosti već postojećih rješenja, pružajući sveobuhvatnu podršku za jednostavno i učinkovito planiranje događaja u manjim, privatnim grupama.

Aplikacija ima širok spektar potencijalnih korisnika. Među glavnim korisnicima su studenti i akademske grupe koje često organiziraju seminare, grupne sastanke ili prezentacije. Timovi unutar tvrtki također mogu imati koristi od ove aplikacije prilikom planiranja sastanaka, radionica ili timskih događaja. Prijatelji i obitelji mogu je koristiti za organizaciju privatnih okupljanja, proslava ili zajedničkih putovanja, dok klubovi i udruge mogu olakšati planiranje svojih redovnih sastanaka, radionica ili volonterskih aktivnosti. Fleksibilnost aplikacije omogućuje njezinu prilagodbu različitim potrebama korisnika, što je čini korisnom za različite vrste događaja, od formalnih poslovnih sastanaka do neformalnih društvenih okupljanja.

Sustav aplikacije uključuje nekoliko važnih funkcionalnosti koje omogućuju lakše planiranje događaja. Prvo, aplikacija nudi korisnicima mogućnost registracije kroz unos korisničkog imena, e-maila i lozinke. Nakon registracije, svi podaci korisnika sigurno se pohranjuju, omogućujući buduću autentifikaciju. Korisnici se zatim mogu prijaviti na svoj račun pomoću korisničkog imena i lozinke te imati pristup svim funkcionalnostima sustava. Nakon prijave, korisnici mogu kreirati nove događaje unosom ključnih detalja poput naslova, datuma, vremena, lokacije i opisa. Ova mogućnost kreiranja objava pomaže korisnicima da pruže važne informacije o događaju na jednom mjestu.

Osim kreiranja događaja, aplikacija omogućuje korisnicima i brisanje objava, čime se osigurava kontrola nad događajima i prilagodba prema potrebama. Nakon što je događaj objavljen, korisnici imaju mogućnost glasanja o predloženim terminima i lokacijama, omogućujući

transparentan način za dogovor. Kreatori događaja, odnosno korisnici koji su objavili određeni događaj, imaju mogućnost odrediti konačno vrijeme i mjesto, čime događaj postaje finalan i zaključan za daljnje promjene. Osim toga, integracija s Google Kalendarom omogućuje korisnicima sinkronizaciju svih događaja, pružajući im sve informacije o događanjima na jednoj platformi, što olakšava planiranje.

Aplikacija nudi i različite opcije interakcije sa svakim događajem. Korisnici mogu označiti svoju prisutnost putem opcije "attend" ili naznačiti da neće prisustvovati putem opcije "won't attend". Uz to, mogu dodati komentare na objave, ostavljajući dodatne informacije ili postavljajući pitanja vezana uz događaj. Aplikacija također podržava predlaganje alternativnih termina ili lokacija, što dodatno potiče suradnju i daje korisnicima veću kontrolu nad organizacijom. Na kraju, korisnici mogu glasati za predložene opcije mjesta i vremena, čime se osigurava demokratičan proces odlučivanja i veće zadovoljstvo svih uključenih.

Opseg projekta uključuje osnovne funkcionalnosti poput kreiranja događaja, glasanja o vremenu i lokaciji, integracije s kalendarom, slanja obavijesti i podsjetnika te mogućnosti komentiranja i interakcije u stvarnom vremenu. Upravljanje korisnicima provodi se kroz sustav autentifikacije i prilagodbu postavki profila, čime aplikacija zadovoljava osnovne korisničke zahtjeve. Ove funkcionalnosti postavljaju temelje za daljnje proširenje i nadogradnje, a budući razvoj može uključivati dodatne opcije za korisničku prilagodbu i analitiku.

Projekt također ima velik potencijal za buduće nadogradnje i proširenja. Među mogućim nadogradnjama su napredna analitika koja bi omogućila praćenje popularnosti događaja, broja prijava i učestalosti sudjelovanja. Dodatno, unaprijeđeni sustav notifikacija mogao bi omogućiti prilagođene podsjetnike prema vremenskim zonama, što bi olakšalo planiranje događaja za međunarodne korisnike. Geolokacijska integracija mogla bi predložiti najbliža mjesta sastanka za sve sudionike, dok bi proširenje podrške na dodatne kalendarske sustave poput Apple Kalendar-a ili Microsoft Outlooka dodatno povećalo dostupnost aplikacije. Na kraju, razvoj mobilne aplikacije poboljšao bi pristup korisnicima, omogućujući im jednostavan pregled događaja i primanje obavijesti na mobilnim uređajima.

+ Add a custom footer

▼ Pages 13

Find a page...

▶ [Home](#)

[1. Opis projektnog zadatka](#)

CognitoAgent /
PlanBot[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

2. Analiza zahtjeva

[Edit](#)[New page](#)[Jump to bottom](#)

Ema Mamić edited this page 1 minute ago · [18 revisions](#)

Funkcionalni zahtjevi

ID zahtjeva	Opis	Prioritet	Izvor	Kriteriji prihvatanja
F-001	Registracija korisnika kreiranjem novog računa pomoću emaila, korisničkog imena i lozinke. Nakon registracije, informacije ostaju sigurno pohranjene i omogućuju buduće autentifikacije.	Visok	Zahtjev dionika	Registracija pomoću emaila, korisničkog imena koji nije zauzeto, lozinke. Dobivaju email o uspješnoj registraciji. Informacije su sigurno pohranjene i koriste se za buduće autentifikacije.
F-002	Mogućnost prijave korisnika na njihov postojeći račun pomoću korisničkog imena i lozinke.	Visok	Zahtjev dionika	Uspješna prijava pomoću korisničkog imena i lozinke omogućuje korištenje platforme i pregled personalnih informacija. Neuspješna autentifikacija prikazuje poruku o neispravnom korisničkom imenu ili lozinki.
F-003	Korisnici imaju mogućnost stvaranja objava s detaljima kao što su naslov, datum, vrijeme, lokacija i opis.	Visok	Zahtjev dionika	Korisnici mogu stvarati nove objave o budućem događaju, pri čemu su ime i opis događaja obavezni, dok su datum i vrijeme neobavezni.
F-004	Korisnici imaju mogućnost brisanja objava s detaljima kao što su naslov, datum, vrijeme, lokacija i opis.	Visok	Zahtjev dionika	Korisnici mogu brisati nove objave o budućem događaju, pri čemu su ime i opis događaja obavezni, dok su datum i vrijeme neobavezni.
				Korisnici mogu svaku objavu označiti s "attend". Na stranici budućih događaja

F-005	Korisnici objavu mogu označiti sa "attend".	Visok	Zahtjev dionika	mogu koristiti filter koji prikazuje samo one objave označene s "attend". Korisnici dobivaju e-mail o novostima vezanim uz označene objave, a oznaku je moguće ukloniti u bilo kojem trenutku.
F-006	Korisnici objavu mogu označiti sa "won't attend".	Visok	Zahtjev dionika	Korisnici mogu svaku objavu označiti s "won't attend". Oznaku je moguće ukloniti u bilo kojem trenutku.
F-007	Korisnici mogu na objavu ostaviti običan komentar.	Srednji	Zahtjev dionika	Korisnik na svaku objavu može ostaviti običan komentar. Svi komentari objave vidljivi su ostalim korisnicima.
F-008	Korisnici na objavu mogu predložiti drugo mjesto.	Visok	Zahtjev dionika	Korisnici za svaku objavu imaju alat kojim mogu predložiti novo mjesto. Prijedlog je vidljiv svim korisnicima.
F-009	Korisnici na objavu mogu predložiti drugo vrijeme.	Visok	Zahtjev dionika	Korisnici za svaku objavu imaju alat kojim mogu predložiti novo vrijeme. Prijedlog je vidljiv svim korisnicima.
F-010	Mogućnost brisanja korisničkog računa.	Visok	Zahtjev dionika	Mogućnost brisanja nekog postojećeg korisničkog računa. Korisnički račun ne mora imati sve potrebne informacije.

Ostali zahtjevi

Zahtjevi za održavanje

ID zahtjeva	Opis	Prioritet
NF-001	Vrijeme odziva sustava za bilo koju radnju (prijava, kreiranje događaja itd.) trebalo bi biti ispod 2 sekunde, uz korištenje optimiziranih upita baze podataka.	Visok
NF-002	Aplikacija bi trebala podržavati sve veći broj korisnika i događaja bez utjecaja na performanse	Visok
NF-003	Sustav mora zaštiti sve osjetljive informacije korisnika (npr. lozinke osobnih podataka). Svi korisnički podaci trebaju biti pohranjeni i preneseni sigurno uz enkripciju	Visok
NF-004	Sustav i korisničko sučelje trebaju biti intuitivni i jednostavnii za korištenje, omogućujući korisnicima da samostalno izvršavaju uobičajene zadatke bez potrebe za dodatnim vodstvom.	Visok
NF-005	Aplikacija bi trebala funkcionirati pouzdano, uz minimalne neplanirane zastoje. U slučaju kvara, sustav bi se trebao oporaviti unutar 15 minuta, čime se osigurava visoka dostupnost i pouzdanost.	Srednji
NF-006	Kod programa trebao bi biti dobro strukturiran i dokumentiran kako bi se olakšalo jednostavno ažuriranje i otklanjanje pogrešaka, čime se omogućuje učinkovito održavanje.	Visok

Dionici

- 1.Korisnici - Krajnji korisnici koji se registriraju, prijavljuju i koriste sustav za organizaciju i praćenje događaja.
- 2.Administratori sustava - Osobe odgovorne za održavanje i nadzor sustava, osiguravajući njegovu dostupnost i sigurnost.
- 3.Naručitelji - Klijenti ili vlasnici projekta koji su financirali i naručili razvoj sustava, odgovorni za smjernice i zahtjeve.
- 4.Razvojni tim - Programeri, dizajneri i inženjeri koji su uključeni u izradu, optimizaciju i ažuriranje aplikacije.



Aktori i njihovi funkcionalni zahtjevi

A-1 Aktor (administrator):

Prijava (F-002) Administrator se prijavljuje pomoću korisničkog imena i lozinke, čime pristupa administracijskim alatima.

Stvaranje i brisanje objava (F-003) Administrator može dodavati nove objave (događaje) i uklanjati ih prema potrebi. Objava može uključivati informacije kao što su naslov, datum, vrijeme, lokacija i opis.

Brisanje objave (F-004) Administrator može obrisati objave, a pritom se brišu svi komentari i prijedlozi vezani za tu objavu.

Označavanje konačnog događaja (F-005) Administrator može postaviti konačne detalje događaja (vrijeme i mjesto), nakon čega događaj postaje finalan i vidljiv na stranici budućih događaja.

Sinkronizacija s Google kalendarom (F-006) Omogućuje dodavanje događaja iz platforme u privatni Google kalendar korisnika.

Obavijesti putem e-maila (F-007) Omogućava slanje e-mail obavijesti korisnicima o novim objavama i budućim događajima.

Označavanje prisustva na događaju ("attend") (F-008) Administrator može označiti događaj kao „attend“, što omogućava korisnicima lakše praćenje njihovih planiranih aktivnosti.

Označavanje neprisustva ("won't attend") (F-009) Administrator može označiti neprisustvo na događajima, što sprječava primanje obavijesti o tim događajima.

Komentiranje objava (F-010) Administrator može dodavati komentare na objave, vidljive svim korisnicima.

Predlaganje alternativnog mjesta i vremena (F-011) Administrator može predložiti drugo mjesto i/ili vrijeme za događaj, omogućujući fleksibilnost u organizaciji.

Glasanje na prijedloge (F-012) Administrator može glasati na predložene lokacije i vrijeme ili ukloniti svoj glas. Svi korisnici vide broj glasova svakog prijedloga.

Brisanje korisničkog računa (F-013) Administrator može ukloniti korisničke račune bez obzira na to jesu li uneseni svi potrebni podaci.

A-2 Aktor (korisnik):

Registracija (F-001) Omogućuje korisniku kreiranje novog računa koristeći e-mail, korisničko ime i

LOZINKU.

Prijava (F-002) Korisnik se može prijaviti na račun koristeći korisničko ime i lozinku, čime dobiva pristup platformi i personaliziranim funkcionalnostima.

Stvaranje i brisanje objava (F-003) Korisnici mogu kreirati i ukloniti vlastite objave, čime organiziraju događaje za druge korisnike.

Označavanje konačnog događaja (F-005) Korisnici koji su kreatori objava mogu definirati konačno mjesto i vrijeme događaja, čime ga postavljaju kao finalan.

Sinkronizacija s Google kalendarom (F-006) Korisnici mogu dodati događaje iz platforme u svoj Google kalendar za lakše praćenje obaveza.

Primanje obavijesti putem e-maila (F-007) Korisnici primaju e-mail obavijesti o novim događajima i objavama unutar nekoliko minuta od njihovog stvaranja.

Označavanje prisustva ("attend") (F-008) Korisnici mogu označiti da će prisustvovati određenom događaju, a na stranici budućih događaja mogu filtrirati događaje označene kao „attend”.

Označavanje neprisustva ("won't attend") (F-009) Omogućava korisnicima označavanje događaja kao „won't attend”, čime prestaju primati obavijesti o tim događajima.

Komentiranje objava (F-010) Korisnici mogu komentirati objave i vidjeti komentare drugih korisnika.

Predlaganje alternativnog mjesta i vremena (F-011) Korisnici mogu predložiti novo mjesto i/ili vrijeme za događaj, čime doprinose organizaciji i zajedničkom dogовору.

Glasanje na prijedloge (F-012) Korisnici mogu glasati na predložena mjesta i vrijeme, pri čemu mogu u bilo kojem trenutku povući svoj glas. Svim je korisnicima vidljiv broj glasova za svaki prijedlog.

Sekundarni aktori

Vanjski sustavi (API, usluge)

Opis: Aplikacija može komunicirati s vanjskim servisima kao što su platni sustavi, e-mail usluge, sustavi za autentifikaciju (OAuth, LDAP) ili drugi web servisi.

Djelatnosti i Primjena funkcionalnosti:

Registracija korisnika i autentifikacija (F-001, F-002): Vanjski sustavi za autentifikaciju poput OAuth-a ili LDAP-a mogu biti korišteni za omogućavanje jednostavne i sigurne prijave korisnika u aplikaciju. Tako korisnici mogu koristiti postojeće virovodnине s drugih platformi, što smanjuje

aplikaciju. Tako korisnici mogu korisiti postojeće vjerodajnice s unutarnjim platformama, što smanjuje potrebu za kreiranjem novog računa. Slanje podsjetnika putem e-pošte: Integracijom s vanjskim e-mail servisima, aplikacija može slati podsjetnike i obavijesti korisnicima o novim objavama i budućim događajima, čime se osigurava pravovremena informiranost. Sinkronizacija s kalendarima (F-006): Kroz integraciju s vanjskim servisima kao što je Google Calendar API, aplikacija omogućuje korisnicima sinkronizaciju događaja s njihovim osobnim kalendarima, pružajući im jedinstveni pregled događaja i obaveza.

Baza podataka

Opis: Baza podataka služi kao skladište podataka aplikacije. Može biti implementirana korištenjem sustava kao što su MySQL, PostgreSQL, MongoDB ili slični.

Djelatnosti i Primjena funkcionalnosti:

Pohrana korisničkih podataka i postavki (F-001, F-002): Baza podataka pohranjuje podatke o korisnicima, uključujući vjerodajnice, postavke profila, raspoloživost i preferencije. Pohrana podataka o događajima (F-003, F-004): Baza podataka omogućava pohranu informacija o događajima, uključujući naslove, datume, vremena, lokacije i opise. Ovi podaci se ažuriraju prema korisničkim akcijama, poput kreiranja, uređivanja ili brisanja događaja. Praćenje glasova i komentara (F-005): Baza podataka pohranjuje sve glasove sudionika vezane za predložene termine i lokacije, kao i komentare i prijedloge korisnika. Na taj način se omogućava demokratično odlučivanje o detaljima događaja i transparentna interakcija.

Sustav za autentifikaciju

Opis: Sustav za autentifikaciju omogućava provjeru identiteta korisnika putem vanjskih servisa kao što su OAuth, OpenID ili lokalni sustavi za prijavu.

Djelatnosti i Primjena funkcionalnosti:

Autentifikacija i provjera identiteta korisnika (F-001, F-002): Sustavi OAuth-a koristi se za autentifikaciju korisnika putem njihovih postojećih računa na drugim platformama (npr. Google, Facebook), smanjujući potrebu za lokalnom prijavom i povećavajući sigurnost. Izdavanje i provjera JWT tokena: Nakon uspješne prijave, sustav generira JWT tokene koji se koriste za autentifikaciju korisnika tijekom sesije. Ti tokeni omogućavaju siguran pristup zaštićenim dijelovima aplikacije i osiguravaju da samo autenticirani korisnici mogu pristupiti osjetljivim informacijama i funkcionalnostima. Sigurnosna kontrola pristupa: Sustav za autentifikaciju osigurava da samo ovlašteni korisnici imaju mogućnost pristupa funkcijama kao što su kreiranje događaja, glasanje i komentiranje, što dodatno poboljšava sigurnost podataka i privatnost korisnika.

CognitoAgent /
PlanBot[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

3. Specifikacija zahtjeva sustava

[Edit](#)[New page](#)[Jump to bottom](#)

Ema Mamić edited this page 12 minutes ago · [10 revisions](#)

Obrasci uporabe

Opis obrazaca uporabe

UC-01: Registriranje

- Glavni sudionik: Neprijavljeni korisnik (novi korisnik)
- Cilj: Kreiranje novog računa.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik nema postojeći račun na platformi.
- Opis osnovnog tijeka:
 1. Korisnik odabire opciju za registriranje
 2. Korisnik unosi svoje osobne podatke (e-mail, korisničko ime, lozinku).
 3. Sustav provodi provjeru validnosti unesenih podataka.
 4. Sustav šalje e-mail o uspješnoj registraciji
- Opis mogućih odstupanja:
 1. Ako korisnik unese neispravan e-mail ili korisničko ime koje već postoji :
 1. Sustav prikazuje poruku o grešci i traži ispravak.
 2. Korisnik ispravlja podatke ili odustaje od prijave
 1. Ako korisnik već ima račun s tom e-mail adresom:
 1. Sustav obavještava korisnika da račun već postoji.

UC-02: Prijava korisnika

- Glavni sudionik: Neprijavljeni korisnik
- Cilj: Omogućiti pristup platformi.
- Sudionici: Korisnik, Baza podataka, OAuth2.0
- Preduvjet: Korisnik je registriran.
- Opis osnovnog tijeka:
 1. Korisnik odabire opciju za prijavu. (F-002)
 2. Korisnik unosi korisničko ime i lozinku.
 3. Sustav provodi provjeru validnosti podataka

3. Sustav prihvati prijavljenu vrijednost podataka.

4. Ako su podaci ispravni, korisnik se uspješno prijavljuje i pristupa aplikaciji.

5. Korisnik je preusmjeren na početnu stranicu

- Opis mogućih odstupanja:

1. Ako korisnik unese pogrešno korisničko ime ili lozinku:

1. Sustav prikazuje obavijest o pogrešnim podacima

UC-03: Prijava korisnika s vanjskom autentifikacijom

- Glavni sudionik: Neprijavljeni korisnik

- Cilj: Omogućiti pristup platformi pomoću vanjske autentikacije

- Sudionici: Korisnik, OAuth2.0

- Preduvjet: Korisnik je registriran.

- Opis osnovnog tijeka:

1. Korisnik odabire opciju za prijavu pomoću vanjske autentifikacije.

2. Sustav preusmjerava korisnika na stranicu vanjske autentifikacije.

3. Korisnik unosi podatke.

4. Ako su podaci ispravni, vanjska autentifikacija šalje potvrdu sustavu.

5. Sustav prijavljuje korisnika

6. Korisnik je preusmjeren na početnu stranicu

- Opis mogućih odstupanja:

1. Ako korisnik unese pogrešne podatke:

1. Sustav prikazuje obavijest o pogrešnim podacima

UC-04: Odjava

- Glavni sudionik: Korisnik

- Cilj: Korisnik se uspješno odjavljuje sa svog korisničkog računa.

- Sudionici: Korisnik

- Preduvjet: Korisnik je prijavljen u sustav.

- Opis osnovnog tijeka:

1. Korisnik odabire opciju za odjavu. (F-003)

2. Sustav potvrđuje odjavu, korisnik gubi pristup aplikaciji.

3. Sustav preusmjerava korisnika na stranicu za prijavu ili početnu stranicu.

UC-05: Kreiranje događaja

- Glavni sudionik: Korisnik
- Cilj: Korisnik kreira novi događaj.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav.
- Opis osnovnog tijeka:
 1. Korisnik odabire opciju za kreiranje događaja. (F-003)
 2. Korisnik unosi detalje o događaju (ime i opis).
 3. Korisnik odabire opciju za dodavanje datuma, vremena i lokacije.
 4. Korisnik potvrđuje unos podataka i klikne na "Spremi događaj".
 5. Sustav sprema događaj i prikazuje ga na listi događaja.
- Opis mogućih odstupanja:
 1. Ako su unosi nepotpuni ili neispravni:
 1. Sustav traži od korisnika da ispravi podatke.
 2. Korisnik ispravlja informacije ili odustaje.
 1. Ako je unesen datum u prošlosti:
 1. Sustav obavještava korisnika i traži da odabere drugi datum.
 2. Korisnik ispravlja datum ili odustaje.

UC-06: Brisanje događaja

- Glavni sudionik: Korisnik
- Cilj: Korisnik briše postojeći događaj.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav i ima pravo brisanja događaja.
- Opis osnovnog tijeka:
 1. Korisnik odabire događaj koji želi obrisati.
 2. Korisnik odabire opciju za brisanje događaja . (F-004)
 3. Sustav potvrđuje akciju brisanja.

4. Sustav vrše dogadjaj (zajedno sa komentarima i prijeaznim iz baze podataka).

UC-07: Finalizacija događaja

- Glavni sudionik: Korisnik
- Cilj: Korisnik finalizira događaj.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je kreirao događaj.
- Opis osnovnog tijeka:
 1. Korisnik odabire događaj koji želi finalizirati.
 2. Korisnik potvrđuje vrijeme i mjesto te finalizira događaj. (F-005)
 3. Sustav označava događaj kao finaliziran i sprječava daljnje izmjene.
- Opis mogućih odstupanja:
 1. Ako korisnik unese neispravni datum ili ne unese sve potrebne informacije:
 1. Sustav traži od korisnika da ispravi podatke.
 2. Korisnik ispravlja informacije ili odustaje.

UC-08: Integracija s Google Kalendarom

- Glavni sudionik: Korisnik
- Cilj: Događaj se automatski dodaje u Google Kalendar kada je finaliziran.
- Sudionici: Korisnik, Baza podataka, Google Kalendar
- Preduvjet: Događaj mora biti finaliziran i korisnik mora biti prijavljen te imati povezan Google račun.
- Opis osnovnog tijeka:
 1. Sustav provjerava je li organizator povezan s Google računom.
 2. Ako je povezan, sustav inicira proces dodavanja događaja u Google Kalendar. (F-006)
 3. Sustav koristi API za Google Kalendar kako bi dodao događaj s podacima (naziv, datum, vrijeme, mjesto).
- Opis mogućih odstupanja:
 1. Ako korisnik nije povezan s Google računom:
 1. Sustav traži od njega da se poveže.

UC-09: Označavanje događaja sa "attend"

- Glavni sudionik: Korisnik
- Cilj: Korisnik označava svoj interes za sudjelovanje na događaju.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen.
- Opis osnovnog tijeka:
 1. Sudionik odabire događaj na kojem želi sudjelovati.
 2. Sudionik klikne na opciju "attend" kako bi označio svoj interes za sudjelovanje. (F-008)
 3. Sustav ažurira status sudionika za taj događaj na "attend".

UC-10: Označavanje događaja sa "won't attend"

- Glavni sudionik: Korisnik
- Cilj: Omogućiti korisnicima opciju da ne žele sudjelovati u događaju.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen.
- Opis osnovnog tijeka:
 1. Korisnik odabire događaj na kojem ne želi sudjelovati.
 2. Korisnik odabire opciju "won't attend" (F-009).
 3. Sustav ažurira status sudionika za taj događaj na "won't attend".

UC-11: Pregled finaliziranih događaja

- Glavni sudionik: Korisnik
- Cilj: Korisnik vidi sve finalizirane događaje.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav.
- Opis osnovnog tijeka:
 1. Korisnik odabire opciju da vidi finalizirane događaje
 2. Korisnik je preusmjeren na stranicu s finaliziranim događajima

UC-12: Filtriranje događaja koje je korisnik označio s "Attend"

- Glavni sudionik: Korisnik
- Cilj: Korisnik filtrira događaje na temelju onih koje je označio kao "attend".

- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav.
- Opis osnovnog tijeka:
 1. Korisnik pristupa popisu svih finaliziranih događaja.
 2. Korisnik koristi filtriranje za odabir događaja na temelju statusa "Attend".
 3. Sustav prikazuje samo one događaje koje je sudionik označio s "Attend".
 4. Korisnik pregledava popis filtriranih događaja.

UC-13: Obavijesti

- Glavni sudionik: Korisnik
- Cilj: Korisnik prima obavijesti za nove događaje, događaje koje je označio kao "attend".
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen.
- Opis osnovnog tijeka:
 1. Sustav prati nove događaje, događaje označene s "attend".
 2. Kada se pojavi novi događaj ili kada se finalizira događaj kojeg je korisnik označio s "attend", sustav šalje obavijest korisniku. (F-007)
 3. Korisnik prima obavijest putem e-maila.

UC-14: Dodavanje komentara

- Glavni sudionik: Korisnik
- Cilj: Korisnik dodaje komentar na odabrani događaj.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen.
- Opis osnovnog tijeka:
 1. Korisnik pristupa događaju kojem želi dodati komentar.
 2. Korisnik odabire opciju za dodavanje komentara. (F-010)
 3. Korisnik unosi komentar u odgovarajuće polje.
 4. Korisnik odabire opciju za slanje komentara.
 5. Sustav prikazuje komentar.
- Opis mogućih odstupanja:

I. AKO KORISNIK pokusa poslati komentar bez teksta:

1. Sustav ne dozvoljava da pošalje prazan tekst.
2. Korisnik može ispraviti ili odustati od slanja komentara.

UC-15: Prijedlog za alternativno mjesto događaja

- Glavni sudionik: Korisnik
- Cilj: Korisnik predlaže alternativno mjesto za događaj.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen i događaj nije finaliziran.
- Opis osnovnog tijeka:
 1. Korisnik pristupa događaju.
 2. Korisnik odabire opciju za prijedlog alternativnog mesta.
 3. Korisnik unosi svoje prijedlog za novo mjesto događaja.
 4. Korisnik šalje prijedlog.
 5. Sustav šalje prijedlog.
- Opis mogućih odstupanja:
 1. Ako korisnik pokuša poslati prijedlog bez teksta:
 1. Sustav ne dozvoljava da pošalje prazan tekst.
 2. Korisnik može ispraviti ili odustati od slanja prijedloga.

UC-16: Prijedlog za alternativno vrijeme događaja

- Glavni sudionik: Korisnik
- Cilj: Korisnik predlaže alternativno vrijeme za događaj.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen i događaj nije finaliziran.
- Opis osnovnog tijeka:
 1. Korisnik pristupa događaju.
 2. Korisnik odabire opciju za prijedlog alternativnog vremena.
 3. Korisnik unosi svoje prijedlog za novo vrijeme događaja.
 4. Korisnik šalje prijedlog.
 5. Sustav šalje prijedlog.

- Opis mogućih odstupanja:
 1. Ako korisnik pokuša poslati prijedlog bez teksta:
 1. Sustav ne dozvoljava da pošalje prazan tekst.
 2. Korisnik može ispraviti ili odustati od slanja prijedloga.

UC-17: Glasanje za alternativno vrijeme ili mjesto

- Glavni sudionik: Korisnik
- Cilj: Omogućiti korisnicima da glasaju za alternativno vrijeme ili mjesto događaja.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen u aplikaciju
- Opis osnovnog tijeka:
 1. Korisnik otvara detalje događaja.
 2. Korisnik bira opciju za glasanje za alternativno vrijeme ili mjesto. (F-013)
 3. Sustav prikazuje opcije koje su već predložene od drugih korisnika.
 4. Korisnik odabire jednu od predloženih opcija.
 5. Sustav evidentira glas korisnika i ažurira ukupni broj glasova za svaku opciju.
- Opis mogućih odstupanja:
 1. Ako alternativno vrijeme ili mjesto nije predloženo:
 1. Korisnik mora odustati od glasanja.

UC-18: Pregled osobnih informacija

- Glavni sudionik: Korisnik
- Cilj: Korisnik pregledava svoje osobne informacije pohranjene u sustavu.
- Sudionici: Korisnik, Baza podataka
- Preduvjet: Korisnik je prijavljen u sustav.
- Opis osnovnog tijeka:
 1. Korisnik pristupa svom korisničkom profilu putem izbornika.
 2. Sustav prikazuje osnovne osobne informacije korisnika (korisničko ime, e-mail).

UC-19: Brisanje korisničkog računa

- Glavni sudionik: Administrator

• Glavni sudionici, administrator

- Cilj: Omogućiti administratoru da trajno izbriše korisnički račun s platforme.

- Sudionici: Administrator, Baza podataka

- Preduvjet: Administrator mora biti prijavljen.

- Opis osnovnog tijeka:

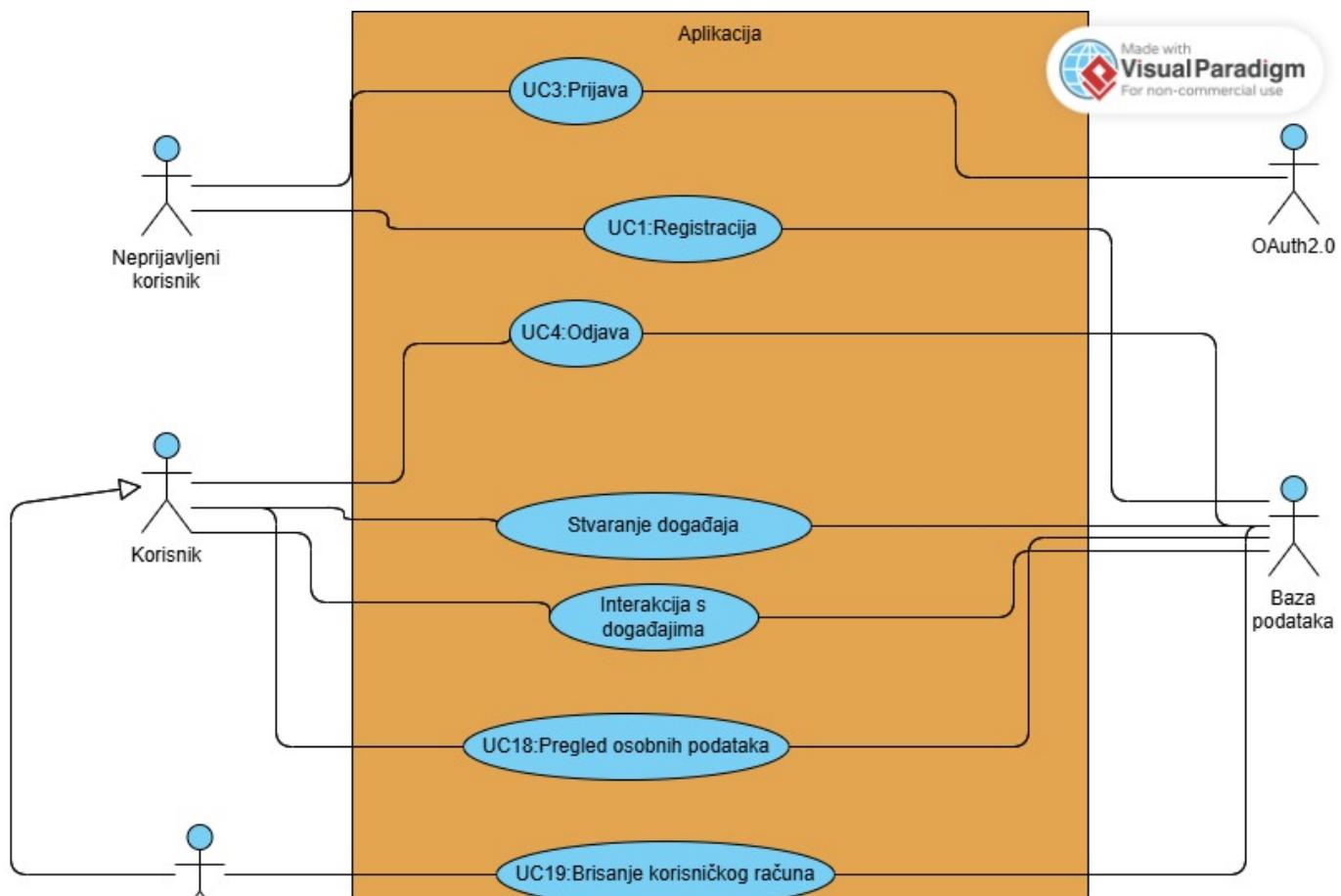
1. Prikaz opcije za brisanje računa (F-014)
2. Administrator odabire opciju "Izbriši račun".
3. Administrator mora potvrditi svoj zahtjev.
4. Sustav briše sve korisnikove podatke.

- Opis mogućih odstupanja:

1. Ako administrator zatvori prozor za potvrdu ili odustane tijekom postupka brisanja:

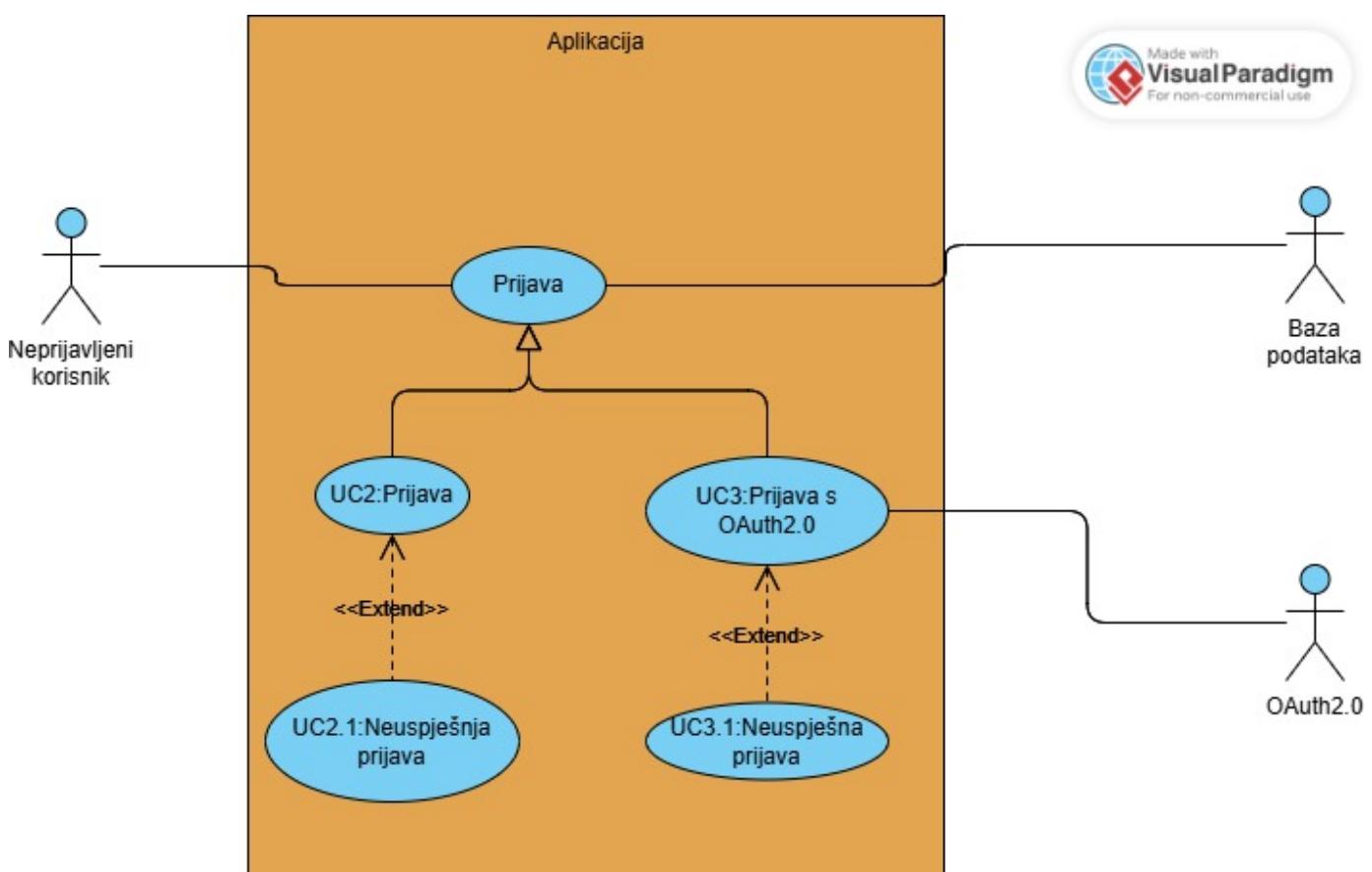
1. Sustav prekida postupak brisanja i korisnik zadržava svoj račun.

Dijagrami obrazaca uporabe

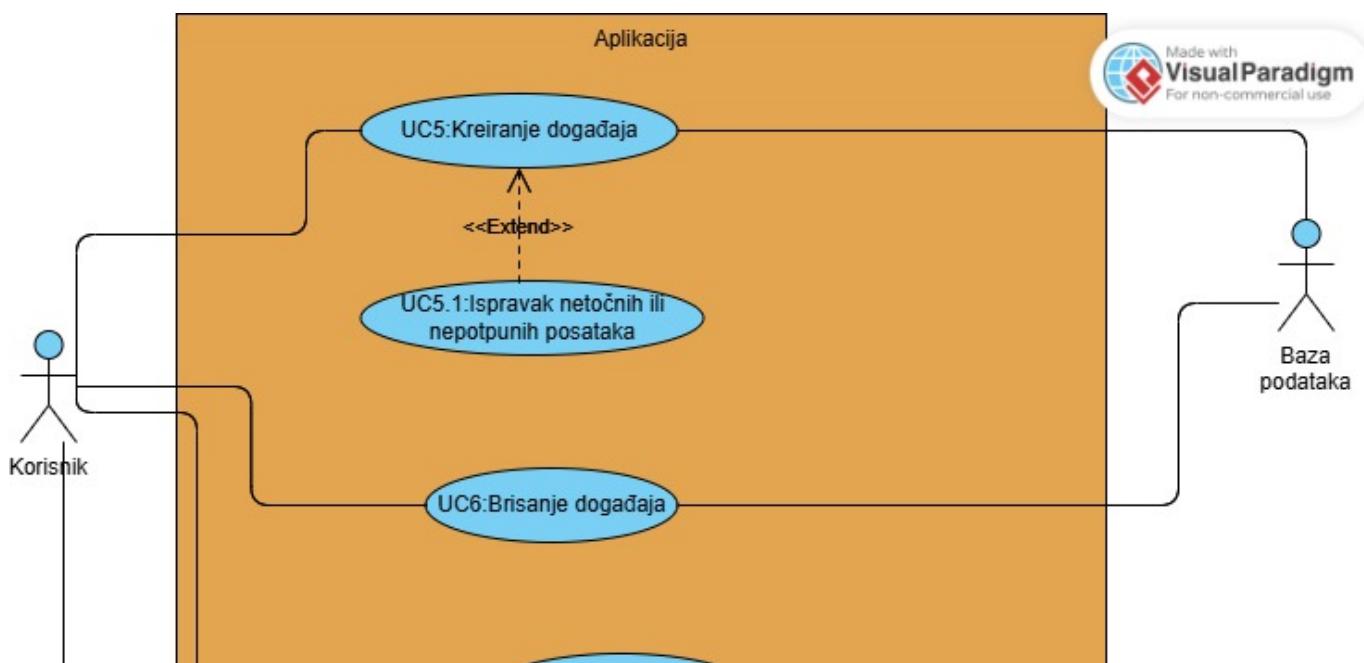


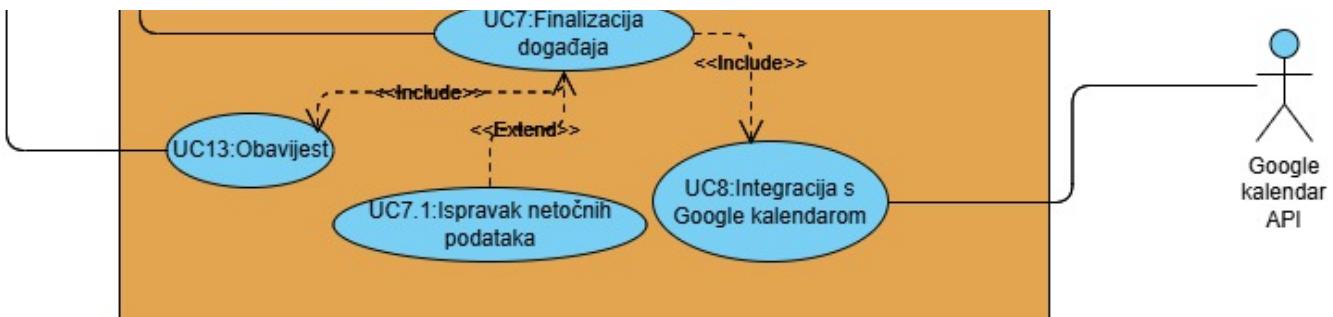


Slika 3.1: Osnovne funkcionalnosti

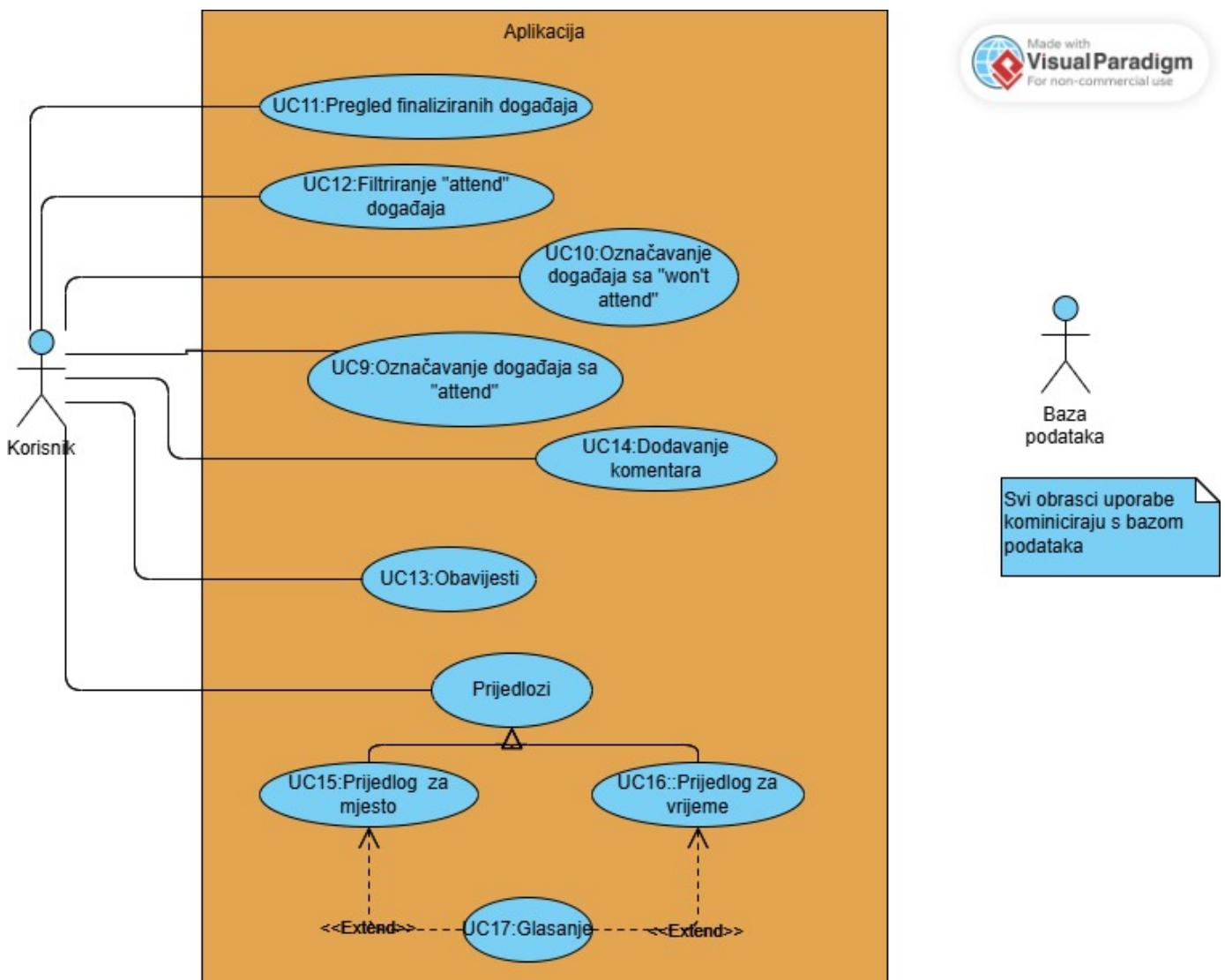


Slika 3.2: Prijava



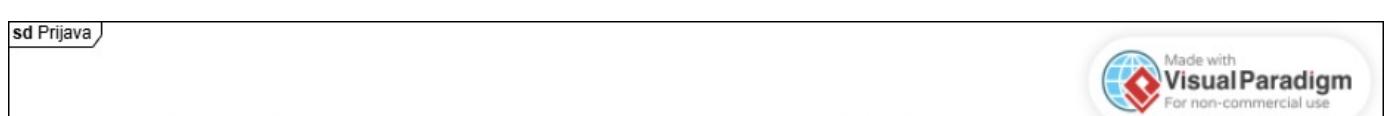


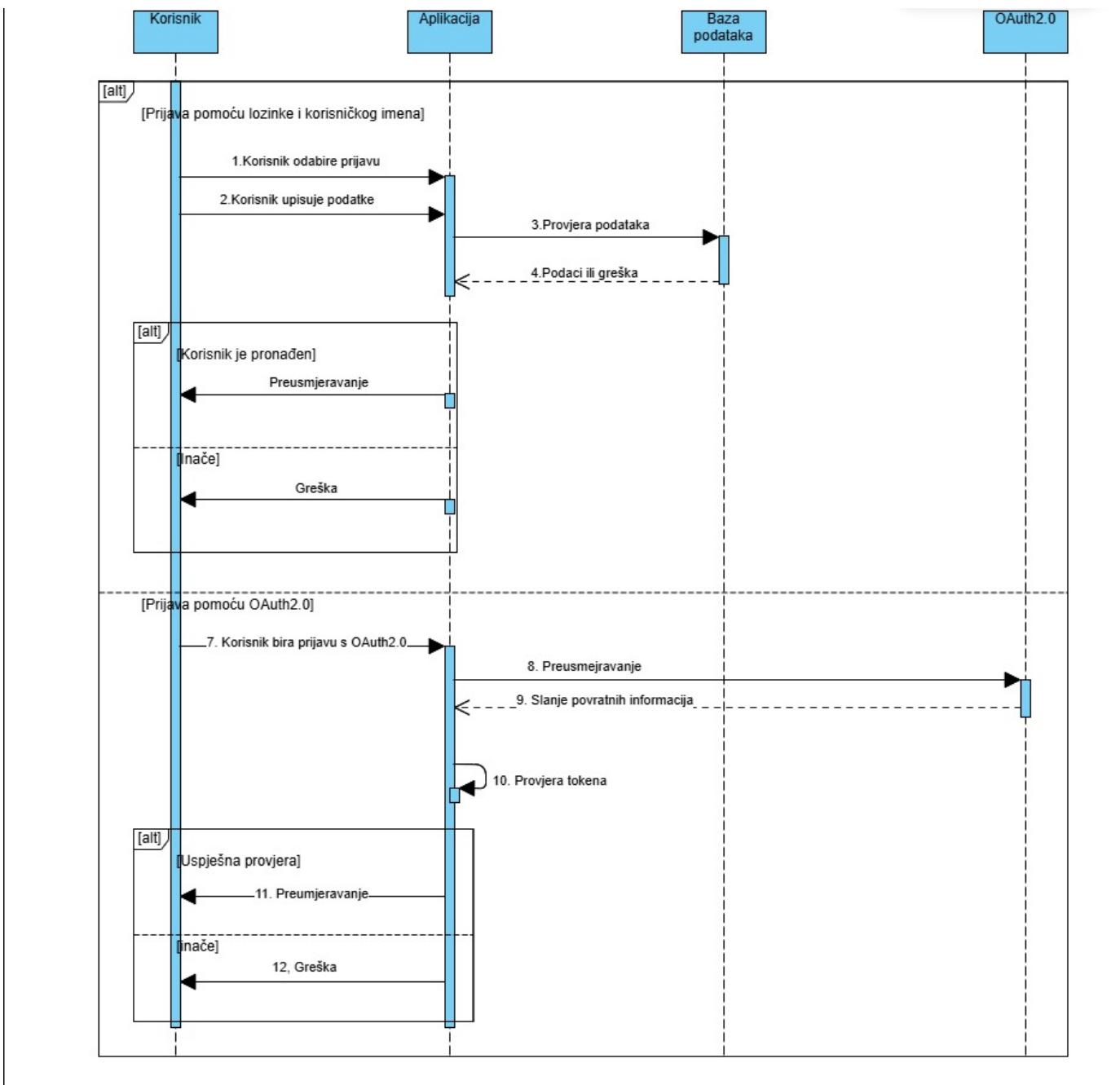
Slika 3.3: Stvaranje događaja



Slika 3.4: Interakcije s događajima

Sekvencijski dijagrami





Slika 3.5 : Prijava u aplikaciju

Sekvensijski dijagram prikazuje proces autentikacije korisnika unutar aplikacije. Korisnik ima dvije opcije za prijavu: standardnu prijavu s korisničkim imenom i lozinkom ili putem vanjske autentifikacije OAuth 2.0.

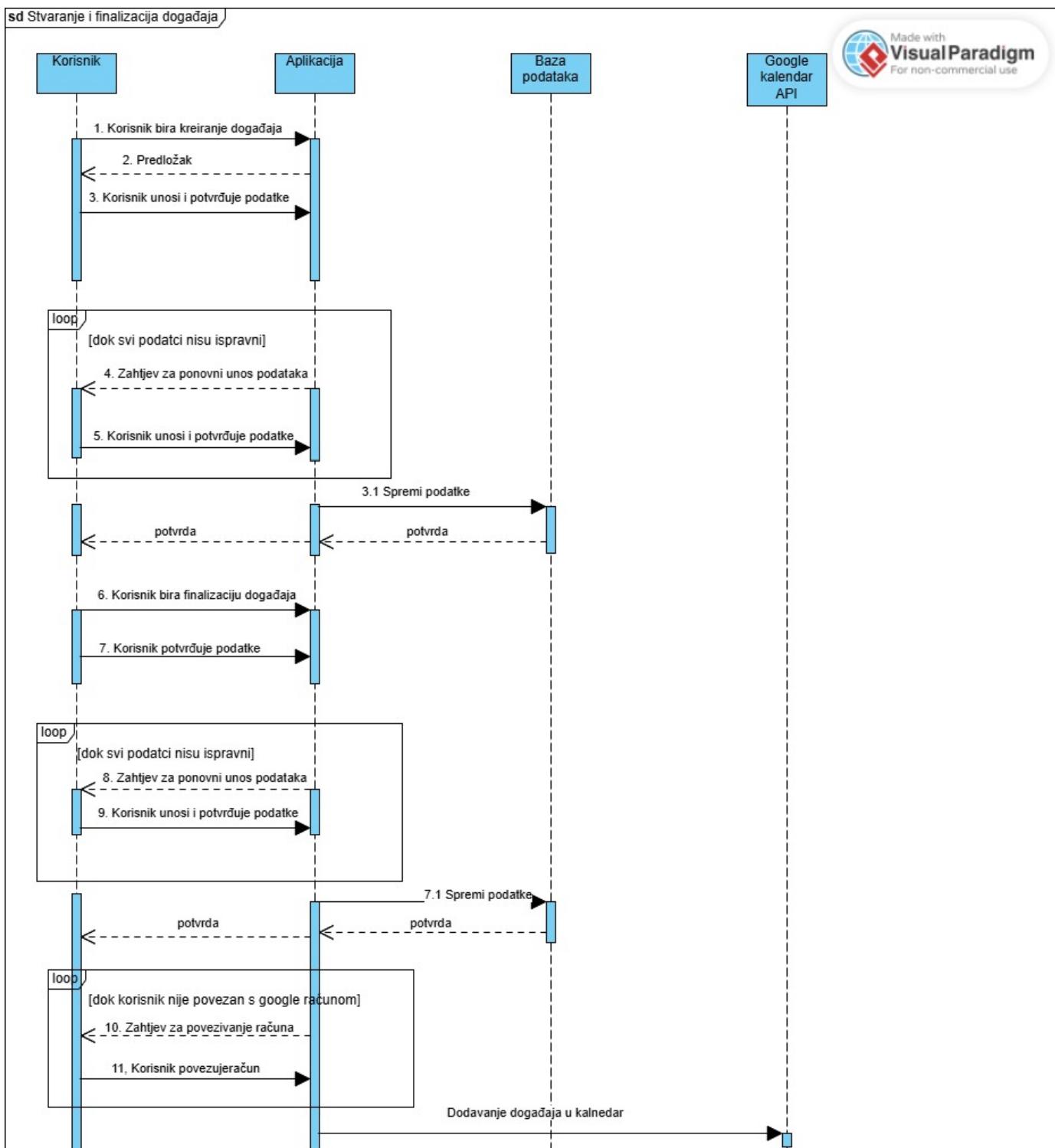
1. Standardna prijava:

- Korisnik unosi svoje korisničko ime i lozinku.
- Sustav provjerava unesene podatke unutar baze podataka.
- Ako su podaci ispravni, korisnik je preusmjeren na početnu stranicu aplikacije.
- U slučaju neispravnih podataka, sustav prikazuje poruku o grešci i omogućava korisniku da ponovno unese podatke.

porinutio unese podatke.

2. Prijava putem OAuth 2.0:

- Korisnik je preusmjeren na sučelje Google OAuth za autentikaciju.
- Nakon uspješne autentikacije na Googleu, korisnik je preusmjeren natrag u aplikaciju na početnu stranicu.
- Ako autentikacija nije uspješna, prikazuje se poruka o grešci i korisniku se nudi mogućnost ponovne prijave.





Slika 3.6 : Kreiranje i finaliziranje događaja

Sekvencijski dijagram ilustrira proces kreiranja i finalizacije novog događaja unutar aplikacije, kao i njegovu integraciju u Google kalendar.

Kada korisnik pokrene postupak kreiranja novog događaja, postoji nekoliko obaveznih i optionalnih koraka koje mora slijediti:

- Unos obaveznih podataka: Korisnik mora unijeti ime i opis događaja. Ovi podaci su ključni za osnovnu identifikaciju i razumijevanje sadržaja događaja.
- Unos optionalnih podataka: Korisnik može dodatno unijeti datum, vrijeme, i lokaciju događaja. Ovi podaci omogućuju bolje planiranje i organizaciju.
- Validacija unesenih podataka: Ako korisnik ne unese obavezne podatke (ime i opis) ili unese datum koji je već prošao, sustav automatski odbija unesene podatke. Sustav zatim javlja grešku i traži od korisnika da korigira ili ponovno unese točne informacije.
- Pohrana događaja u bazi podataka: Kada sustav verificira i prihvati sve unesene podatke kao valjane, šalje ih u bazu podataka. Događaj se zatim pohranjuje i automatski prikazuje na listi svih nadolazećih događaja u aplikaciji.

Kada korisnik pristupi opciji za finalizaciju događaja, potrebno je izvršiti nekoliko ključnih koraka kako bi se osigurala ispravnost i spremanje podataka događaja:

- Potvrda vremena i mjesta događaja: Prije finalizacije, korisnik mora potvrditi ili ponovno unijeti vrijeme i mjesto održavanja događaja kako bi se osigurala njihova točnost.
- Provjera i validacija unesenih podataka: Ako korisnik ne unese sve potrebne podatke ili ako unese datum koji je već prošao, sustav automatski detektira ove pogreške. Sustav zatim obavještava korisnika o grešci te zahtijeva ponovni unos ispravnih i ažuriranih podataka.
- Finalizacija događaja: Nakon što korisnik unese sve potrebne i ispravne podatke, sustav procedira s finalizacijom događaja. Ovaj korak uključuje spremanje potvrđenih podataka u bazu podataka te zaključavanje događaja za daljnje izmjene.

Nakon finalizacije događaja, sustav pokreće proces integracije događaja u Google kalendar koristeći Google kalendar API. Postupak se odvija u nekoliko koraka:

- Pokušaj spremanja događaja u Google kalendar: Sustav automatski pokušava dodati finalizirani događaj u Google kalendar korisnika, koristeći unesene podatke o događaju kao

sto su vrijeme, datum i mjesto.

- Provjera povezanosti Google računa: Ako korisnikov Google račun nije povezan s aplikacijom, sustav detektira nedostatak autorizacije. Korisniku se tada prikazuje obavijest s uputom da poveže svoj Google račun kako bi omogućio sinkronizaciju događaja.
- Povezivanje s Google računom: Korisnik slijedi upute za povezivanje svog Google računa, što uključuje odobrenje pristupa aplikacije potrebnim informacijama iz njegovog Google računa. Nakon uspješne autorizacije, sustav automatski nastavlja s dodavanjem događaja u kalendar.

Provjera uključenosti ključnih funkcionalnosti u obrasce uporabe

Obrazac uporabe	Funkcijski zahtjev
UC-1	F-001
UC-2	F-002
UC-3	F-002
UC-4	-
UC-5	F-003
UC-6	F-004
UC-7	F-005
UC-8	F-006
UC-9	F-008
UC-10	F-009
UC-11	-
UC-12	-
UC-13	F-007
UC-14	F-010
UC-15	F-011
UC-16	F-012
UC-17	-

UC-17	F-013
UC-18	-
UC-19	F-014

+ Add a custom footer

▼ Pages 13

Find a page...

▶ Home

▶ 1. Opis projektnog zadatka

▶ 2. Analiza zahtjeva

▼ 3. Specifikacija zahtjeva sustava

Obrasci uporabe

Opis obrazaca uporabe

UC-01: Registriranje

UC-02: Prijava korisnika

UC-03: Prijava korisnika s vanjskom autentifikacijom

UC-04: Odjava

UC-05: Kreiranje događaja

UC-06: Brisanje događaja

UC-07: Finalizacija događaja

UC-08: Integracija s Google Kalendarom

UC-09: Označavanje događaja sa "attend"

UC-10: Označavanje događaja sa "won't attend"

UC-11: Pregled finaliziranih događaja

UC-12: Filtriranje događaja koje je korisnik označio s "Attend"

UC-13: Obavijesti

UC-14: Dodavanje komentara

UC-15: Prijedlog za alternativno mjesto događaja

CognitoAgent /
PlanBot[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

4. Arhitektura i dizajn sustava

[Edit](#)[New page](#)[Jump to bottom](#)

Ema Mamić edited this page 15 minutes ago · [17 revisions](#)

Arhitektura sustava

Opis arhitekture

Stil Arhitekture

Odabrana arhitektura je slojevita (ili n-tier) s MVC (Model-View-Controller) obrascem na backendu. Ova arhitektura temelji se na korištenju Spring Boota za backend i Reacta za frontend. Ovakav slojeviti pristup ima nekoliko prednosti. Prvo, omogućuje jasnu raspodjelu odgovornosti između različitih dijelova sustava. U ovom slučaju, React upravlja prezentacijskim slojem (View), dok Spring Boot organizira poslovnu logiku (Controller) i pristup podacima (Model). Time se osigurava neovisno upravljanje frontendom i backendom, što poboljšava održivost sustava i omogućava brže uočavanje i ispravljanje eventualnih problema.

Nadalje, slojevita arhitektura omogućuje visoku razinu fleksibilnosti. Na primjer, ako se poveća opterećenje sustava ili dodaju nove značajke, svaki sloj se može optimizirati i prilagoditi zasebno, neovisno o drugima. Tako se backend može poboljšati dodatnom optimizacijom servera ili baze podataka, dok frontend može koristiti rješenja poput load balancinga ili cachinga. Ova arhitektura također osigurava visoku razinu sigurnosti. Korištenjem JWT (JSON Web Tokens) autentifikacije backend može sigurno provoditi autentifikaciju korisnika bez izlaganja osjetljivih informacija, dok se kontrola pristupa lako primjenjuje na različite izvore i akcije unutar sustava.

Razdvajanje frontend i backend slojeva omogućava lakše održavanje, budući da se svaki sloj može neovisno nadograđivati, ažurirati i prilagođavati bez utjecaja na druge slojeve. Osim toga, modularnost ove arhitekture olakšava buduće proširenje i omogućuje jednostavnu integraciju dodatnih usluga poput Google karte ili vanjskih API-ja.

Podsustavi Programa

Sustav se sastoji od nekoliko ključnih podsustava koji zajedno omogućuju osnovne funkcionalnosti aplikacije. Prvi podsustav je upravljanje korisnicima, koji uključuje sve korisničke akcije kao što su registracija i autentifikacija. Proces registracije obrađuje UserService, dok se autentifikacija provodi pomoću JWT tokena. Pri tome JwtFilter provjerava valjanost tokena prije nego što se korisniku omogući pristup, a MyUserDetailsService provjerava postoji li korisnik u bazi pomoću korisničkog imena.

Drugi podsustav je upravljanje događajima, koji omogućuje stvaranje i upravljanje objavama. Ovaj dio sustava obuhvaća PostService, koji upravlja zahtjevima za stvaranje ili brisanje postova, te PostRepo, koji pretražuje objave iz baze podataka. Podrzana je i funkcionalnost komentiranja objava te predlaganja termina ili lokacija za događaje.

Sigurnost je osigurana kroz posebne podsustave za autentifikaciju i validaciju tokena korisnika. JWTService generira JWT tokene za autentificirane korisnike, dok JwtFilter provjerava valjanost tokena za pristup razlicitim stranicama i funkcionalnostima aplikacije. Upravljanje API-jem omogućuje interakciju između frontenda i backenda putem REST API-ja.

Posljednji važan podsustav je upravljanje podacima, koji se bavi svim interakcijama s bazom podataka i odgovoran je za pohranu i dohvatanje podataka. Ovaj dio sustava koriste UserRepo za podatke o korisnicima te PostRepo za podatke o objavama.

Preslikavanje na Radnu Platformu

Backend je razvijen pomoću Spring Boota kao Maven projekt u Eclipseu, a poslužitelj na backend strani koristi različite servise za glavne funkcionalnosti aplikacije, poput registracije korisnika, prijave i upravljanja događajima. Korisnički zahtjevi dolaze na backend putem UserLoginControllera, dok JWT i sigurnosne usluge omogućuju autentifikaciju i autorizaciju korisnika putem JwtFilter i SecurityConfig.

Baza podataka PostgreSQL koristi se za pohranu podataka o korisnicima i događajima, a UserRepo i PostRepo omogućuju komunikaciju s bazom za potrebe pohrane i dohvatanja podataka o korisnicima i objavama. Na strani klijenta, frontend je izgrađen pomoću Reacta, razvijenog u Visual Studio Codeu. Korisničko sučelje React aplikacije šalje HTTP zahtjeve prema backendu za akcije poput registracije, prijave i kreiranja događaja te prikazuje dobivene odgovore.

Komunikacijski Protokoli

Komunikacija između frontenda i backenda odvija se putem HTTP/HTTPS protokola kako bi se osigurala sigurnost. Backend koristi REST API za komunikaciju, pri čemu izlaže RESTful krajnje točke koje frontend može koristiti za pristup i manipulaciju podacima potrebnim za funkcionalnosti registracije, prijave i kreiranja događaja.

Razvojno i Testno Okruženje

U razvoju se koristi lokalna konfiguracija koja omogućuje pokretanje svih komponenti lokalno. PostgreSQL radi kao lokalna baza podataka, backend usluge su konfiguirane u Eclipseu, dok React frontend radi u Visual Studio Codeu. Git i GitHub koriste se za verzioniranje koda, dok je Eclipse namijenjen upravljanju backend dijelom, a Visual Studio Code frontend dijelom. Za testiranje komunikacije i ispravnost odgovora backend usluga koristi se alat Postman.

Spremište Podataka

Podaci se pohranjuju u bazi podataka PostgreSQL, a struktura baze obuhvaća nekoliko tablica. Tablica users pohranjuje podatke o korisnicima, uključujući korisničko ime, lozinku (šifriranu

pomoću BCrypta) i adresu e-pošte. Tablica posts čuva informacije o objavama, dok tablica image_attachment pohranjuje slike povezane s objavama. Tablica comments služi za pohranu komentara vezanih uz objave, a suggestions za prijedloge termina i lokacija. Relacije između ovih tablica omogućuju povezivanje objava s komentarima i prijedlozima te s korisnicima koji prate ili ne prate određene objave.

Baza podataka

Za ovaj projekt odlučili smo se koristiti PostgreSQL bazu podataka. PostgreSQL je objektno-relacijska baza podataka, što znači da kombinira mogućnosti relacijskih baza podataka, kao što su tablice, redovi i stupci, s dodatnim mogućnostima rada koristeći objektne tipove. Implementacija: Kreirali smo PostgreSQL server koristeći službene pakete dostupne za naš operativni sustav. Nakon instalacije, kreirali smo bazu podataka koristeći pgAdmin. Kreirali smo potrebne sheme i tablice unutar baze podataka. Dodali smo početne podatke u tablice i kreirali indekse za optimizaciju upita. Svaki atribut identifikatora (id) je auto-generated pa smo za njih napravili sequences. Na kraju smo integrirali PostgreSQL bazu podataka s našom Java Spring Boot aplikacijom koristeći JPA/Hibernate i konfigurirali sve potrebne postavke u application.properties datoteci.

Opis tablica

users

Atribut	Tip podatka	Opis varijable
id	BIGINT	Jedinstveni identifikator korisnika
username	VARCHAR	Korisničko ime
password	VARCHAR	Korisnička lozinka
email_address	VARCHAR	Korisnička email adresa

Tablica users pohranjuje informacije o korisnicima sustava, uključujući njihove osnovne podatke poput korisničkog imena, lozinke i email adrese. Uloga ove tablice je omogućiti autentifikaciju korisnika i njihovu identifikaciju unutar aplikacije kako bi mogli pristupati i koristiti sustav.

posts

Atribut	Tip podatka	Opis varijable
...	BIGINT	...

id	BIGINT	Jedinstveni identifikator objave
post_number	BIGINT	Broj objave
status	VARCHAR	Status objave
picture_id	BIGINT	Identifikator slike
published_by_id	BIGINT	Identifikator korisnika
date	DATE	Datum objave
description	VARCHAR	Opis objave
location	VARCHAR	Odabrana lokacija na objavi
title	VARCHAR	Naslov
published_by	VARCHAR	Ime korisnika

Tablica posts služi za pohranu svih objava koje korisnici stvaraju. Svaka objava ima svoj naslov, opis, lokaciju i datum, a povezana je s korisnikom koji ju je stvorio. Ova tablica predstavlja temeljni sadržaj aplikacije.

image_attachment

Atribut	Tip podatka	Opis varijable
id	BIGINT	Jedinstveni identifikator slike
image_desc	BIGINT	Opis slike
imageurl	VARCHAR	URL slike

Tablica image_attachment koristi se za spremanje podataka o slikama koje su pridružene objavama. Ime ulogu dodavanja vizualnog sadržaja uz objave.

comments

Atribut	Tip podatka	Opis varijable
id	BIGINT	Jedinstveni identifikator komentara
text	VARCHAR	Tekst komentara
from_user_id	BIGINT	Identifikator korisnika koji je napisao komentar

Tablica comments nohranijuje komentare koje korisnici pišu na određene objave. Svaki komentar

je povezan s korisnikom koji ga je napisao i s objavom na koju se odnosi. Ona omogućuje korisnicima međusobnu interakciju i diskusiju unutar aplikacije.

posts_comments

Atribut	Tip podatka	Opis varijable
post_id	BIGINT	Identifikator objave
comments_id	BIGINT	Identifikator komentara

Tablica posts_comments djeluje kao poveznica između objava i komentara. Njena uloga je omogućiti pridruživanje više komentara pojedinoj objavi, pri čemu svaki komentar pripada jednoj objavi.

posts_suggestions

Atribut	Tip podatka	Opis varijable
post_id	BIGINT	Identifikator objave
suggestions_id	BIGINT	Identifikator prijedloga na objavu

Tablica posts_suggestions služi za povezivanje prijedloga korisnika s objavama. Svrha je omogućiti davanje novih prijedloga.

suggestions

Atribut	Tip podatka	Opis varijable
id	BIGINT	Jedinstveni identifikator prijedloga
location	VARCHAR	Lokacija prijedloga na objavu
time	TIMESTAMP	Predloženo vrijeme
from_user_id	BIGINT	Identifikator korisnika koji je napisao prijedlog

Tablica suggestions pohranjuje unesene prijedloge. Svaki prijedlog sadrži informacije poput lokacije, vremena i korisnika kao kreatora tog događaja. Ova tablica omogućuje korisnicima pružanje povratnih informacija ili sugestija svake objave.

user_joined_posts

Atribut	Tip podatka	Opis varijable
post_id	BIGINT	Identifikator objave
user_id	BIGINT	Identifikator korisnika

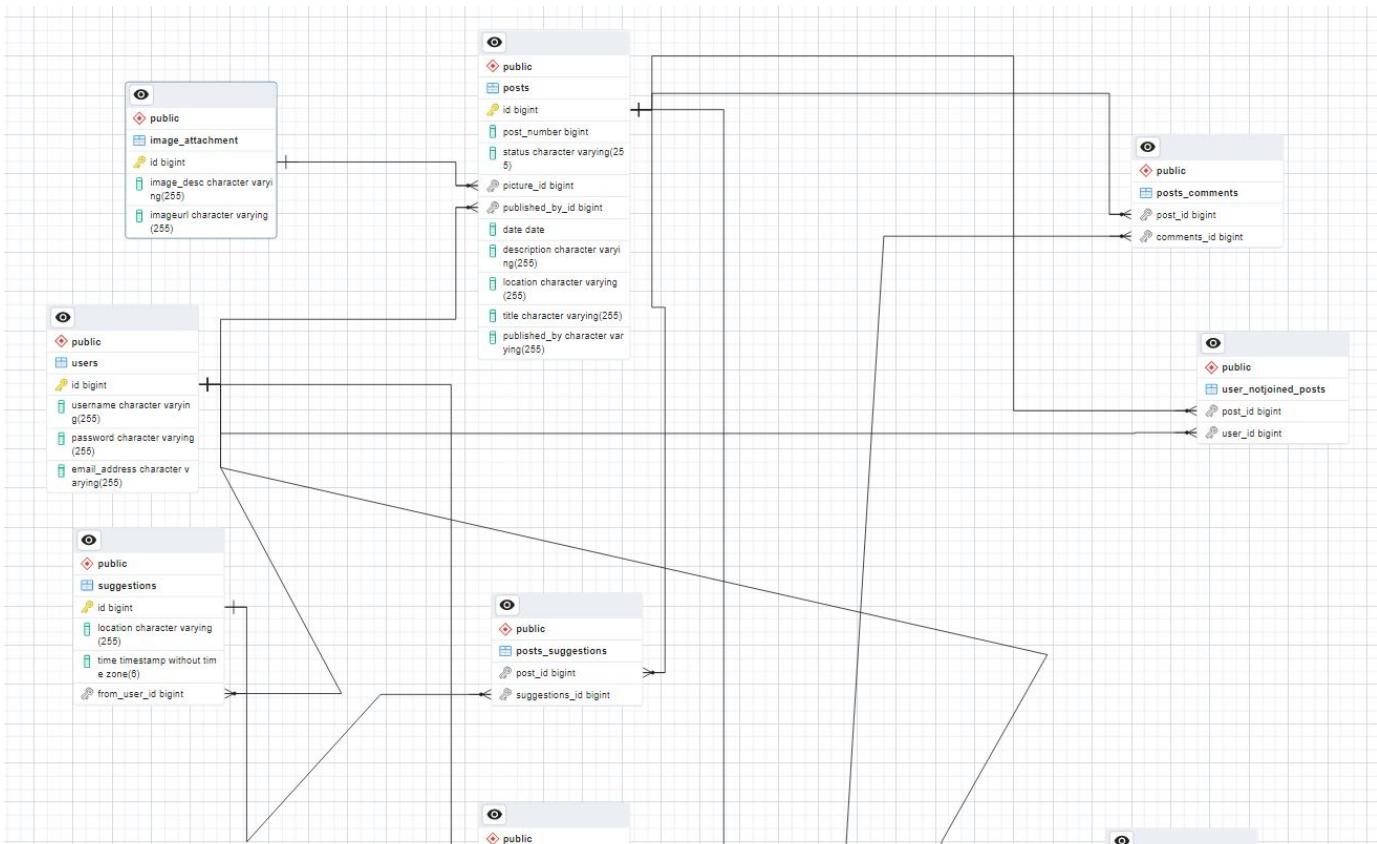
Tablica user_joined_posts prati korisnike koji su se pridružili određenim objavama ili događajima. Njena uloga je evidentiranje aktivnosti korisnika kako bi se znalo tko je sudjelovao u nekoj objavi ili sadržaju.

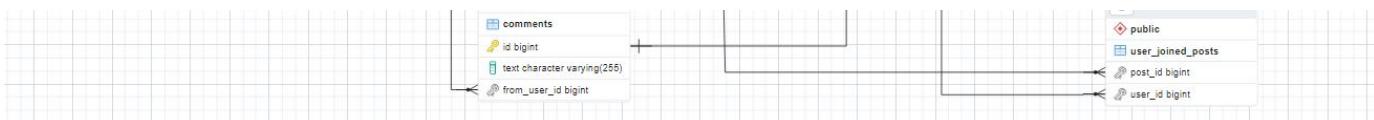
user_notjoined_posts

Atribut	Tip podatka	Opis varijable
post_id	BIGINT	Identifikator objave
user_id	BIGINT	Identifikator korisnika

Tablica user_notjoined_posts ima ulogu praćenja korisnika koji se nisu pridružili određenim objavama. Ova tablica služi za analizu korisničkih aktivnosti i može se koristiti za informiranje ili podsjećanje korisnika o sadržaju koji nisu pregledali ili na kojem nisu sudjelovali.

Dijagram baze podataka





slika 4.1: Dijagram baze podataka

Dijagram baze podataka prikazuje strukturu baze podataka razvijene za sustav planiranja događaja. Baza podataka je normalizirana kako bi se smanjila redundantnost podataka te eliminirale nepoželjne karakteristike poput anomalija umetanja, ažuriranja i brisanja.

Uključuje sljedeće tablice:

1. **posts** – Glavna tablica za spremanje informacija o objavama kao što su naziv, status, datum, vrijeme itd. Tablica sadrži primarni ključ id i strane ključeve koji povezuju druge entitete.
2. **image_attachment** – Služi za pohranu reference na sliku objave.
3. **users** – Tablica koja sadrži podatke o korisnicima, kao što su korisničko ime, e-mail, lozinka
4. **suggestions** – Tablica koja bilježi prijedloge vezane uz objave.
5. **post_suggestions** – Pomoćna tablica koja povezuje prijedloge (suggestions) s objavama (posts).
6. **comments** – Sadrži komentare korisnika. Povezana je sa korisnikom i objavom.
7. **user_joined_posts** – Povezuje korisnike s objavama na kojima sudjeluju.
8. **user_notjoined_posts** – Povezuje korisnike s objavama koje su odbili.
9. **posts_comments** – Pomoćna tablica koja povezuje objave (posts) i komentare (comments).

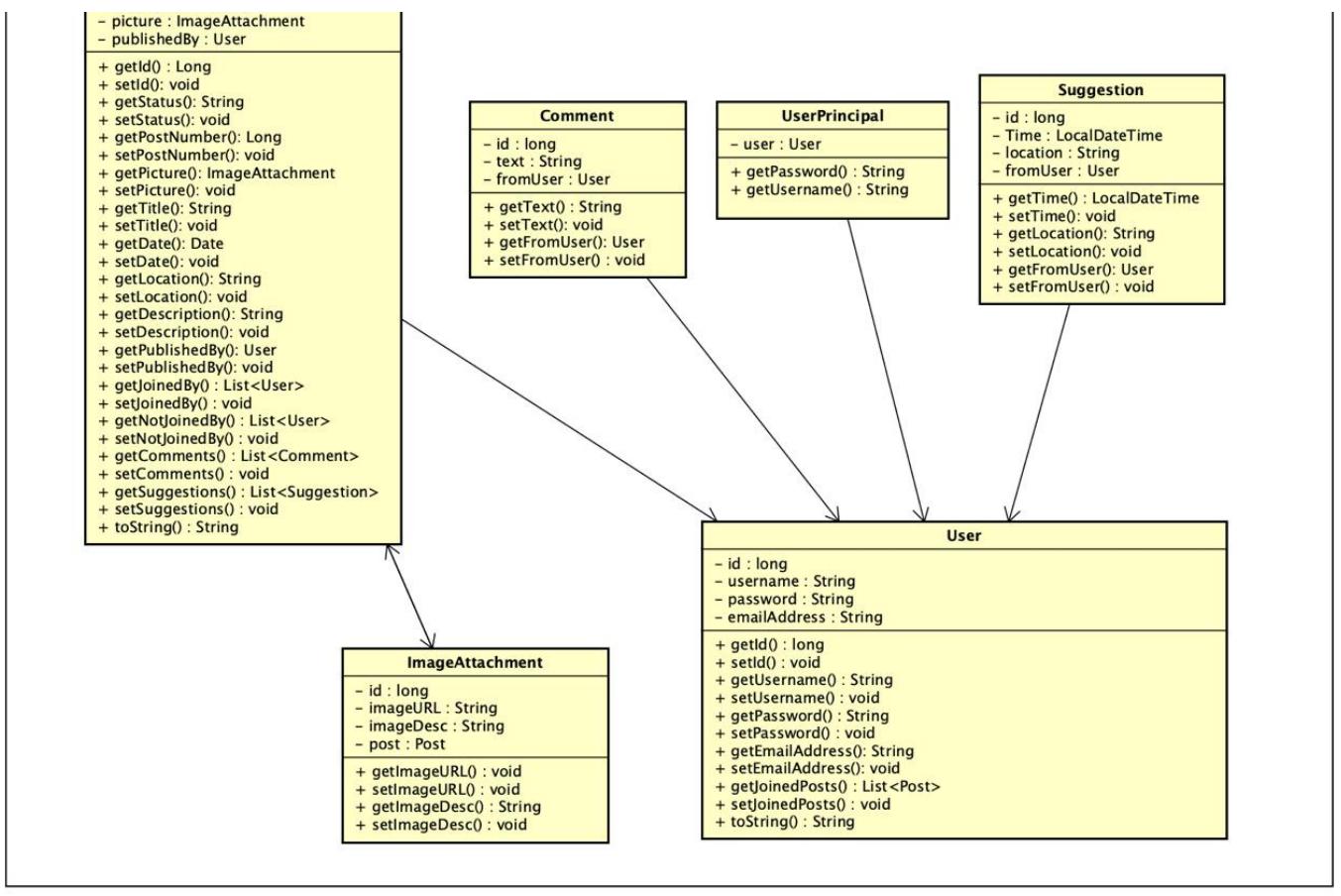
Svi primarni i strani ključevi su jasno označeni u dijagramu, a tablice su povezane relacijama 1:N i M:N prema potrebama sustava. Na primjer:

- users je povezana s posts putem relacije 1:N jer svaki korisnik može objaviti više objava.
- posts i comments su povezane pomoću tablice posts_comments kako bi se podržala fleksibilna veza između više objava i komentara.

Normalizacija: Baza podataka je normalizirana do treće normalne forme (3NF). Svaki stupac sadrži atomske (nedjeljive) vrijednosti, svi neključni atributi u potpunosti ovise o svim ključnim atributima te su neovisni jedan o drugome.

Dijagram razreda

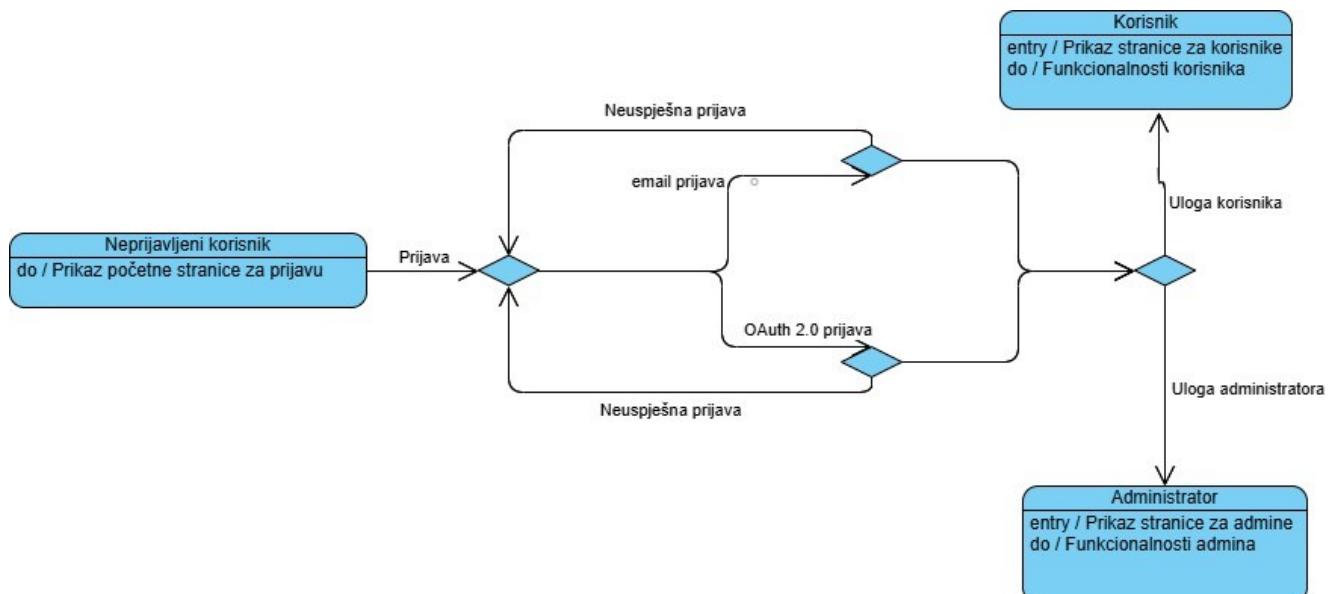




slika 4.2: Dijagram razreda

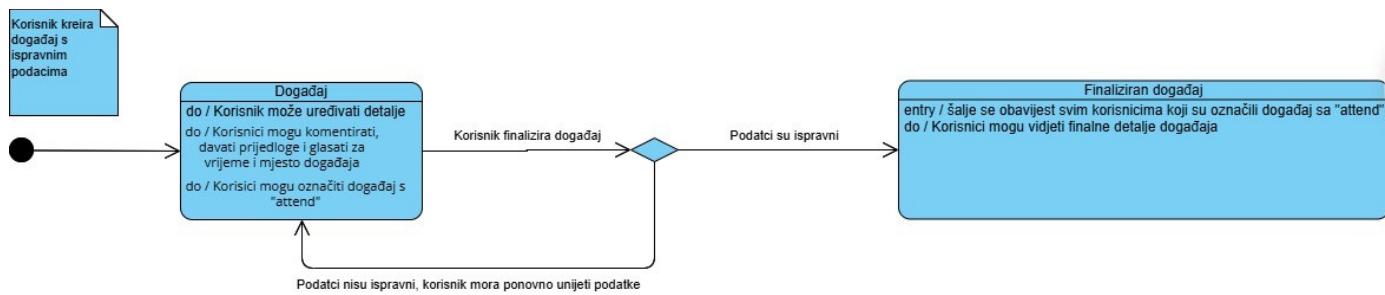
Dinamičko ponašanje aplikacije

UML dijagrami stanja



slika 4.3: Stanja korisnika

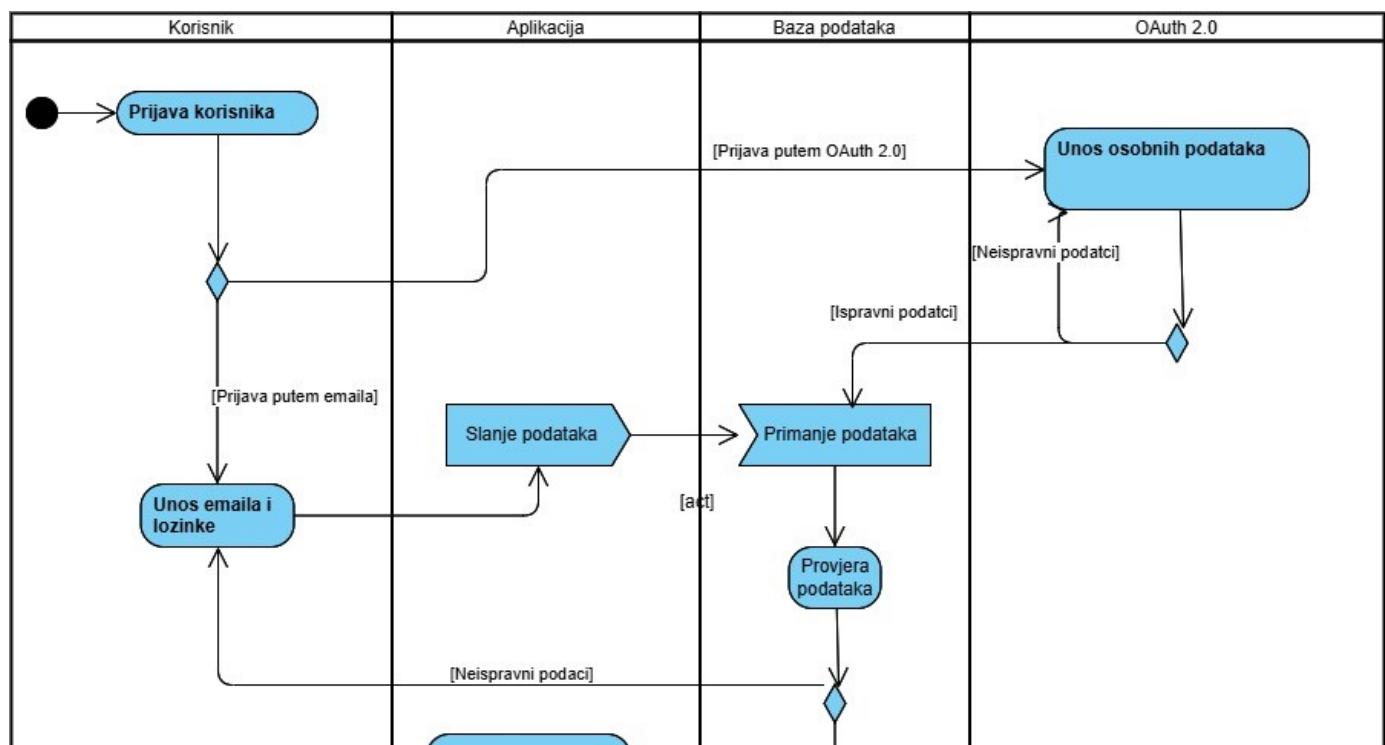
Dijagram prikazuje stanja korisnika. Početno stanje je Neprijavljeni korisnik, u ovom stanju korisnik ima pristup samo početnoj stranici za prijavu i registraciju. Korisnik se prijavljuje emailom i lozinkom, ili s OAuth 2.0 autentifikacijom. Neispravna prijava vraća korisnika na stranicu za prijavu i on ostaje neprijavljen. Uspješna prijava daje korisniku pristup aplikaciji u ulozi Administratora ili običnog Korisnika.



slika 4.4: Stanja događaja

Dijagram prikazuje stanja događaja. Kada korisnik kreira događaj s ispravnim podatcima, događaj se nalazi u početnom stanju. Drugi korisnici mogu vidjeti događaj, označiti ga s "attend", komentirati, davati prijedloge za alternativno vrijeme i mjesto. Korisnik može uređivati detalje događaja i kada korisnik finalizira događaj sa svim podatcima, aplikacija provjerava ispravnost podataka. Ako je neki od podataka neispravan, vraća se u stanje Događaj,a ako su svi podaci ispravni, događaj je finaliziran i prelazi u Finaliziran događaj.

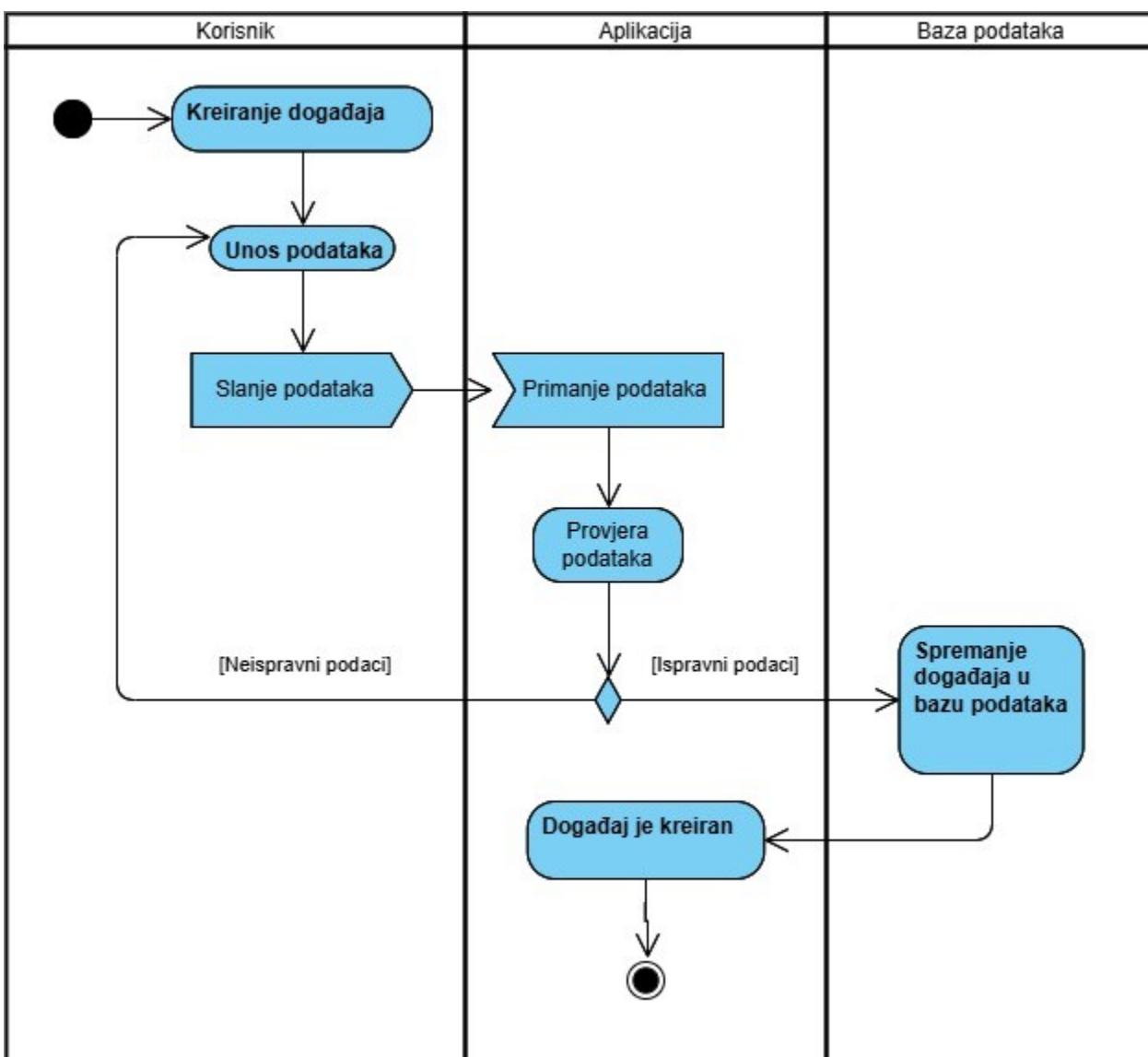
UML dijagrami aktivnosti





slika 4.5: Dijagram aktivnosti prijave

Dijagram prikazuje prijavu korisnika u aplikaciju. Korisnik odabire prijavu emailom i lozinkom ili prijavu s OAuth 2.0. Kod prijave s emailom, podatci se šalju bazi podataka na provjeru. Ako su podatci ispravni, korisniku se daje pristup aplikaciji, a ako su podatci neispravni vraća ga se na početnu stranicu. Kod OAuth 2.0 prijave, korisnik je preusmjeren na vanjsku autentifikaciju gdje se provjerava ispravnost podataka. Ako su podatci ispravni, korisnik dobiva pristup aplikaciji.



slika 4.6: Dijagram aktivnosti tijekom kreiranja događaja

Dijagram prikazuje kreiranje događaja. Korisnik unosi podatke, podatci se šalju i aplikacija ih provjerava. Ako su podatci neispravni, korisnik ponovno mora unijeti podatke. Ako su podatci ispravni, događaj se spremi u bazu podataka, kreiran je i prikazuje se na stranici događaja.

+ Add a custom footer

▼ Pages 13

Find a page...

▶ [Home](#)

▶ [1. Opis projektnog zadatka](#)

▶ [2. Analiza zahtjeva](#)

▶ [3. Specifikacija zahtjeva sustava](#)

▼ [4. Arhitektura i dizajn sustava](#)

- Arhitektura sustava
 - Opis arhitekture
 - Baza podataka
 - Opis tablica
 - users
 - posts
 - image_attachment
 - comments
 - posts_comments
 - posts_suggestions
 - suggestions
 - user_joined_posts

instanci povezan je Load Balancer koji pomoću certifikata omogucava i usmjerava HTTPS zahtjeve iz frontenda prema backendu. Backend aplikacija izrađena je u Spring Boot frameworku i zadužena je za obradu poslovne logike, autentifikaciju korisnika te komunikaciju s bazom podataka. Backend poslužitelj također koristi HTTP/HTTPS protokol za razmjenu podataka s frontendom i bazom podataka.

Baza podataka smještena je na Aiven Cloud Platformi. Unutar baze pohranjuju se svi podaci aplikacije, uključujući korisničke račune, postavke i ostale informacije. Struktura baze definirana je kroz PostgreSQL shemu koja uključuje tablice, relacije i upite. Backend poslužitelj pristupa bazi podataka kako bi dohvatio ili mijenjao potrebne podatke.

Cijela arhitektura koristi sigurne protokole (HTTPS) za zaštitu komunikacije između slojeva, a opisani sustav prati standardni trostupanjski model: frontend (korisničko sučelje), backend (logika aplikacije) i baza podataka (trajna pohrana).

+ Add a custom footer

▼ Pages 13

Find a page...

- ▶ [Home](#)
- ▶ [1. Opis projektnog zadatka](#)
- ▶ [2. Analiza zahtjeva](#)
- ▶ [3. Specifikacija zahtjeva sustava](#)
- ▶ [4. Arhitektura i dizajn sustava](#)
- ▼ [5. Arhitektura komponenata i razmještaja](#)
 - Dijagram komponenata
 - Dijagram razmještaja

- ▶ [6. Ispitivanje programskog rješenja](#)
- ▶ [7. Tehnologije za implementaciju aplikacije](#)
- ▶ [8. Upute za puštanje u pogon](#)
- ▶ [9. Zaključak i budući rad](#)
- ▶ [A. Popis literature](#)
- ▶ [B. Dnevnik promjena dokumentacije](#)
- ▶ [C. Prikaz aktivnosti grupe](#)

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>





Code

Issues 1

Pull requests

Actions

Projects

Wiki

Security

6. Ispitivanje programskog rješenja

[Edit](#)[New page](#)[Jump to bottom](#)

Borna Maričak edited this page 2 hours ago · [2 revisions](#)

Ovo poglavlje treba opisati provedena ispitivanja implementiranih funkcionalnosti na razini komponenti i sustava. Fokus je na odabiru i izvedbi ispitnih slučajeva koji obuhvaćaju redovne, rubne uvjete i testiranje grešaka, kao i upotrebu odgovarajućih alata za provedbu testiranja.

Ispitivanje komponenti

Ovo poglavlje opisuje ispitivanja osnovnih funkcionalnosti (komponenti) naše web-aplikacije, uz prikaz redovnih, rubnih i pogrešnih scenarija. Kao alat za testiranje korišten je **Selenium** unutar okruženja **Python/PyTest**.

Zadaci

1. Razviti minimalno 6 ispitnih slučajeva koji obuhvaćaju:

- **Redovne slučajeve:** uobičajeno (očekivano) ponašanje.
- **Rubne uvjete:** provjera ponašanja na granici valjanosti.
- **Izazivanje pogreške (exception throwing):** kako komponenta reagira na iznimke.
- **Nepostojeće funkcionalnosti:** poziv funkcije ili rute koja nije implementirana.

2. Svaki ispitni slučaj treba sadržavati:

- Opis testirane funkcionalnosti.
- Ulazne podatke.
- Očekivane rezultate.
- Dobivene rezultate (je li test prošao ili pao).
- Opis postupka ispitivanja (korak po korak).

3. Izvorni kod ispitnih slučajeva nalazi se u repozitoriju (test datoteka `test_planbot.py`).

1. Ispitni slučaj: `test_login_success`

Funkcionalnost

Prijava korisnika s ispravnim vjerodajnicama.

- **Ulaz**
 - Korisničko ime: `user1`
 - ⋮

- Lozinka: password
- Očekivani rezultat
 - Aplikacija uspješno logira korisnika te ga preusmjerava na /adminpanel .
- Dobiveni rezultat
 - Test prošao (otvara se URL sadrži "/adminpanel").
- Postupak ispitivanja
 - i. Otvoriti /login .
 - ii. Unijeti user1 i password.
 - iii. Kliknuti Sign in.
 - iv. Provjeriti sadrži li trenutni URL /adminpanel .

```
`def login(driver, username="user1", password="password"): """Pomoćna funkcija za login s  
točnim ili netočnim podacima.""" driver.get(f"{BASE_URL}/login")`
```



```
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[1]/  
input').send_keys(username)  
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[2]/  
input').send_keys(password)  
  
sign_in_button = driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/input')  
sign_in_button.click()`
```

```
`def test_login_success(driver): """Test: točan login (user1/password) -> očekujemo uspjeh i  
prelazak na adminpanel.""" login(driver, "user1", "password")`
```



```
time.sleep(3)  
  
current_url = driver.current_url  
assert "/adminpanel" in current_url, \  
f"Login nije uspio, ostali smo na: {current_url}"`
```

2. Ispitni slučaj: test_login_failure

Funkcionalnost

Prijava korisnika s pogrešnom lozinkom.

- Ulaz
 - Korisničko ime: user1

- Lozinka: `wrong_password`
- **Očekivani rezultat**
 - Aplikacija prikazuje poruku/popup "Login failed".
- **Dobiveni rezultat**
 - Test *prošao* (detektiran je JavaScript `alert` s "Login failed").
- **Postupak ispitivanja**
 - i. Otvoriti `/login`.
 - ii. Unijeti `user1` i `wrong_password`.
 - iii. Kliknuti **Sign in**.
 - iv. Provjeriti je li prikazan `alert` s tekstrom "Login failed".

```
`def test_login_failure(driver): """Test: netočan login -> očekujemo popup ili poruku 'Login failed'."""
    login(driver, "user1", "wrong_password")
```

```
time.sleep(3)
try:
    # Neki popup (alert) - Selenium ga prepoznaje kao alert:
    alert = driver.switch_to.alert
    alert_text = alert.text
    print("Alert je pronađen, tekst:", alert_text)
    assert "Login failed" in alert_text, "Nije pronađen očekivani tekst 'Login failed' u popupu"
    # Klik na OK
    alert.accept()
except:
    # Ako nema alert, možda se prikazuje error unutar stranice
    # (Ako postoji recimo element <div id="errorMsg">Login failed</div>)
    pass`
```



4. Ispitni slučaj: `test_create_event`

Funkcionalnost

Kreiranje novog događaja na `/adminpanel` te provjera prikaza na `/publishedevents`.

- **Ulaz**
 - Naziv događaja: "Test Event Selenium"
 - Datum: "2025-01-24"
 - Lokacija: "Zagreb"

- Opis: "Ovo je test event kreiran Seleniumom."

- **Očekivani rezultat**

- Aplikacija prikaže alert "Event created successfully!"
- Novi se događaj pojavljuje u listi objavljenih na /publishedevents .

- **Dobiveni rezultat**

- Test *prošao*.

- **Postupak ispitivanja**

- i. Prijaviti se kao admin.

- ii. Otvoriti /adminpanel , ispuniti formu (title, date, location, description).

- iii. Kliknuti **Create Event**.

- iv. Potvrditi alert "Event created successfully!".

- v. Otvoriti /publishedevents , pronaći novi event.

```
`def test_create_event(driver): """Test kreiranja novog događaja na /adminpanel i provjera prikaza na /publishedevents.""" login(driver, "user1", "password") time.sleep(1)
```

```
# 1. Otići na /adminpanel (gdje je forma za kreiranje eventa)
driver.get(f"{BASE_URL}/adminpanel")
time.sleep(2) # Kratko čekanje da se stranica učita

# 2. Ispuniti formu:
# - Title
driver.find_element(By.XPATH, '//*[@id="root"]/div/div[2]/form/label[1]/input').send_keys("Test Event Selenium")

# - Date: za <input type="date" name="date" ...> obično vrijedi format "yyyy-mm-dd"
date_input = driver.find_element(By.XPATH, '//*[@id="root"]/div/div[2]/form/label[2]/input')
date_input.clear()
date_input.send_keys("2025-01-24") # pun format za date input

# - Location
driver.find_element(By.XPATH, '//*[@id="root"]/div/div[2]/form/label[3]/input').send_keys("Zagreb")

# - Description
driver.find_element(By.XPATH, '//*[@id="root"]/div/div[2]/form/label[4]/textarea').send_keys(
    "Ovo je test event kreiran Seleniumom."
)

time.sleep(1)

# 3. Klik na "Create Event"
```



```
# 3. Klik id create event
create_btn = driver.find_element(By.XPATH, '//*[@id="root"]/div/div[2]/form/input')
create_btn.click()

try:
    WebDriverWait(driver, 5).until(EC.alert_is_present())
    alert = driver.switch_to.alert
    assert "Event created successfully!" in alert.text, "Unexpected alert text"
    alert.accept() # Click OK on the alert
except TimeoutException:
    pytest.fail("Alert 'Event created successfully!' nije prikazan.")

# 4. Pričekati da backend završi obradu i da se novi event pojavi
# Prema screenshotu, nakon kreiranja se NIJE automatski preusmjerilo na /publishedevents,
# pa ćemo mi ručno otići na /publishedevents.
time.sleep(2)
driver.get(f"{BASE_URL}/publishedevents")

# 5. Sada provjeriti da se "Test Event Selenium" nalazi u objavljenim događajima.
# (Ako se dogovoreno prikazuje upravo na /publishedevents.)
time.sleep(2) # Po potrebi zamijenite s explicit wait
titles = driver.find_elements(By.XPATH, '//*[@id="root"]/div/div[3]/div/h3')
all_titles = [t.text for t in titles]
print("Trenutni naslovi na /publishedevents:", all_titles)

assert "Test Event Selenium" in all_titles, (
    "Novi event 'Test Event Selenium' nije pronađen na /publishedevents!"
)`
```

5. Ispitni slučaj: test_delete_event

Funkcionalnost

Brisanje postojećeg događaja i provjera da li nestaje s popisa na /publishedevents i /eventlist .

- **Ulaz**
 - Ulogirani admin, postojeći event na /publishedevents .
- **Očekivani rezultat**
 - JavaScript alert/popup (npr. "Post deleted successfully!").
 - Event više **nije** vidljiv.
- **Dobiveni rezultat**
 - Test *prošao*.

- Postupak ispitivanja

- Prijaviti se kao admin.
- Otvoriti /publishedevents , kliknuti "Delete" na prvom eventu.
- Potvrditi alert (ako postoji).
- Refresh i provjeriti da event više nije prisutan ni na /publishedevents ni na /eventlist .

```
`def test_delete_event(driver): """Test: Brisanje nekog postojećeg eventa na /publishedevents i provjera na /eventlist.""" # 1. Login login(driver, "user1", "password") time.sleep(3)
```

```
# 2. Open /publishedevents
driver.get(f"{BASE_URL}/publishedevents")
time.sleep(3)

# 3. Locate all events
event_containers = driver.find_elements(By.XPATH, "//div[contains(@style, 'border: 1px solid')]")
if not event_containers:
    pytest.skip("Nema događaja za brisanje.")

# 4. Extract the title of the first event
first_event = event_containers[0]
event_title = first_event.find_element(By.TAG_NAME, "h3").text

# 5. Find and click the "Delete" button for the first event
delete_button = first_event.find_element(By.XPATH, "./button[text()='Delete']")
delete_button.click()
time.sleep(2)

try:
    WebDriverWait(driver, 5).until(EC.alert_is_present())
    alert = driver.switch_to.alert
    print("Popup text:", alert.text)
    alert.dismiss() # Dismiss the alert
except TimeoutException:
    print("No alert appeared after clicking Delete.")

# 7. Refresh /publishedevents and verify the event is deleted
driver.get(f"{BASE_URL}/publishedevents")
time.sleep(3)

remaining_titles = [
    event.find_element(By.TAG_NAME, "h3").text
    for event in driver.find_elements(By.XPATH, "//div[contains(@style, 'border: 1px solid')]")
]
assert event_title not in remaining_titles, "Event i dalje postoji na /publishedevents
...`
```



(nije obrisan)."

```
# 8. Navigate to /eventlist to verify the event is removed there as well
driver.get(f"{BASE_URL}/eventlist")
time.sleep(3)

eventlist_titles = [
    event.find_element(By.TAG_NAME, "h3").text
    for event in driver.find_elements(By.XPATH, "//div[contains(@style, 'display: flex; flex-direction: column;')]")]
]
assert event_title not in eventlist_titles, "Event i dalje postoji na /eventlist (nije obrisan).``
```

Ispitni slučaj: test_register_and_login_as_non_admin

Ulazi:

- Novi korisnik npr. testuser42 (username), test42@example.com (email), pass123 (lozinka).

Koraci ispitivanja:

1. Otvoriti /register , unijeti podatke (testuser42 ...), klik "Register".
2. Očekivano: prelazi na /adminpanel .
3. Otvoriti /login , unijeti testuser42 , pass123 . Klik "Sign in".
4. Kliknuti "Admin view" → očekuje se "You are not an admin" alert.

Očekivani izlaz:

- Uspješna registracija, uspješan login, ali pri "Admin view" → "You are not an admin".

Dobiveni izlaz:

- Test prošao (uvidom u alert).

```
`def test_register_and_login_as_non_admin(driver):
    """ 1. Registrira novog korisnika na /register. 2. Provjeri da li je nakon registracije redirectan na /adminpanel (uspješna registracija). 3. Pokušaj se odjaviti (ili direktno testiraj) i prijaviti s novim korisnikom (provjera logina). 4. Pokušaj pristupa /adminview -> očekuje se "You are not an admin".
    """
    # ====== KORAK 1: REGISTRACIJA ======
    driver.get(f"{BASE_URL}/register")
    time.sleep(1)
```

```
# (Za test, generiraj nasumičan username ili stalan, npr. "testuser42")
new_username = "testuser42"
```



```
new_email = "test42@example.com"
new_password = "pass123"

# Popuni polja:
# username
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[1]/input').send_keys(new_username)
# email
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[2]/input').send_keys(new_email)
# password
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[3]/input').send_keys(new_password)
# confirm password
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[4]/input').send_keys(new_password)

# Klik na "Register"
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/input').click()
time.sleep(2)

# ====== KORAK 2: PROVJERA REDIRECT NA /adminpanel ======
current_url = driver.current_url
assert "/adminpanel" in current_url, f"Registration failed, current_url={current_url}"

# ====== KORAK 3: LOGOUT (ako potrebno), zatim LOGIN s novim userom ======
# (Ovisno o vašoj aplikaciji - ako postoji poseban logout button, kliknite ga;
# inače jednostavno idemo na /login.)
driver.get(f"{BASE_URL}/login")
time.sleep(1)

# Polja za login (username, password)
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[1]/input').send_keys(new_username)
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/label[2]/input').send_keys(new_password)

# Klik "Sign in"
driver.find_element(By.XPATH, '//*[@id="root"]/div/div/form/input').click()
time.sleep(2)

# Ovisno o vašoj aplikaciji, možda se logira odmah na /adminpanel ili /eventlist.
# Provjera:
current_url = driver.current_url
assert "/adminpanel" in current_url or "/eventlist" in current_url, \
    f"Login s novim userom nije uspio, current_url={current_url}"

# ====== KORAK 4: POKUŠAJ PRISTUPA /adminview I OČEKIVANJE 'You are not an admin' ======
```

```

driver.find_element(By.XPATH, '//*[@id="root"]/div/div[1]/button[3]').click()

time.sleep(2)
time.sleep(3)

# Ako se pojavi popup alert:
try:
    alert = driver.switch_to.alert
    alert_text = alert.text
    print("ALERT TEXT:", alert_text)
    assert "You are not an admin" in alert_text, "Očekivana poruka 'You are not an admin' nije pronađena!"
    alert.accept()
except:
    # Ako nema alert, možda se prikazuje error unutar stranice
    # Prilagodite selektor i tekst koji se prikazuje
    body_text = driver.find_element(By.TAG_NAME, "body").text
    assert "You are not an admin" in body_text, "Nije pronađena poruka 'You are not an admin' ni u body!"`
```

Naziv testa: `test_navigacija_stranica` Opis: Nakon uspješne prijave, ovaj test provjerava ispravnost navigacije između nekoliko ključnih stranica (Admin panel, Admin view, Published events, Event list) pritiskom na odgovarajuće gume u sučelju. 1. Prijava: • Koristimo pomoćnu funkciju `login(driver, "user1", "password")` koja otvara /login i unosi vjerodajnice. • Nakon uspješne prijave, možemo pristupiti gumbima za navigaciju. 2. Koraci ispitivanja: 1. Klik na "New Event" → Očekuje se prelazak na /adminpanel. 2. Klik na "Admin View" → Očekuje se prelazak na /adminview. 3. Klik na "Published Events" → Očekuje se prelazak na /publishedevents. 4. Klik na "Event List" → Očekuje se prelazak na /eventlist. 3. Očekivani rezultat: • Kod svakog klika, URL treba sadržavati odgovarajući dio puta (npr. /adminpanel, /adminview, /publishedevents, /eventlist). • Ako se stranica ne promijeni kako je predviđeno, test pada s odgovarajućom porukom ("Nije otvorena ..."). 4. Dobiveni rezultat: • Kod ispravne implementacije, svi aserti prolaze i test uspijeva. 5. Zaključak: radi • Ovaj test pomaže provjeriti jesu li gumbi za navigaciju funkcionalni i vode li zaista na predviđene rute (stranice).

```
`def test_navigacija_stranica(driver):
    """ Test: Nakon logina, navigacija između stranica pritiskom na gume i provjera da li se otvaraju ispravno.
    """
    login(driver, "user1", "password")
    time.sleep(3)
```

```
# 1. Klik na "New Event" button to navigate to /adminpanel
new_event_button = driver.find_element(By.XPATH, '//*[@id="root"]/div/div[1]/button[1]')
new_event_button.click()
time.sleep(2)
assert "/adminpanel" in driver.current_url, "Nije otvorena adminpanel stranica"
```



```
# 2. Klik na "Admin View" button to navigate to /adminview
admin_view_button = driver.find_element(By.XPATH, '//*[@id="root"]/div/div[1]/button[3]')
admin_view_button.click()
time.sleep(2)
assert "/adminview" in driver.current_url, "Nije otvorena adminview stranica"

# 3. Klik na "Published Events" button to navigate to /publishedevents
published_events_button = driver.find_element(By.XPATH, '//*[@id="root"]/div/div[1]/button[2]')
published_events_button.click()
time.sleep(2)
assert "/publishedevents" in driver.current_url, "Nije otvorena publishedevents stranica"

# 4. Klik na "Event List" button to navigate to /eventlist
event_list_button = driver.find_element(By.XPATH, '//*[@id="root"]/div[2]/button[1]')
event_list_button.click()
time.sleep(2)
assert "/eventlist" in driver.current_url, "Nije otvorena eventlist stranica"
```

+ Add a custom footer

▼ Pages 13

Find a page...

▶ [Home](#)

▶ [1. Opis projektnog zadatka](#)

▶ [2. Analiza zahtjeva](#)

▶ [3. Specifikacija zahtjeva sustava](#)

▶ [4. Arhitektura i dizajn sustava](#)

▶ [5. Arhitektura komponenata i razmještaja](#)

▼ [6. Ispitivanje programskog rješenja](#)

Ispitivanje komponenti

Zadaci

- 2. Ispitni slučaj: test_login_failure
- 4. Ispitni slučaj: test_create_event
- 5. Ispitni slučaj: test_delete_event
- Ispitni slučaj: test_register_and_login_as_non_admin

▶ [7. Tehnologije za implementaciju aplikacije](#)

▶ [8. Upute za puštanje u pogon](#)

▶ [9. Zaključak i budući rad](#)

▶ [A. Popis literature](#)

▶ [B. Dnevnik promjena dokumentacije](#)

▶ [C. Prikaz aktivnosti grupe](#)

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>



[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

7. Tehnologije za implementaciju aplikacije

[Edit](#)[New page](#)[Jump to bottom](#)

Lea Gojić edited this page 3 weeks ago · [3 revisions](#)

Korištene tehnologije i alati

1. Korišteni programski jezici:

U svrhu razvoja ove aplikacije za pozadinski dio korišten je Java 21.0.3. Razlog tome je dugogodišnja reputacija Java kao stabilne, sigurne i pouzdane platforme za veće projekte. Omogućava održivost projekta i jednostavnu integraciju s velikim brojem dostupnih biblioteka. U izradi korisničkog sučelja upotrijebljen je JavaScript 20.17. Ova verzija pruža kompatibilnost s modernim alatima i lako se prilagođava brzim promjenama u svijetu web razvoja, što značajno pridonosi učinkovitom frontend razvoju.

2. Radni okviri i biblioteke:

Za poslužiteljsku stranu aplikacije odabran je Spring Boot 3.3.5, zajedno s pripadajućim starter paketima Spring Boot Starter Data JPA, Spring Boot Starter Security, Spring Boot Starter Web i Spring Boot Starter Test. Primjena Spring Boota omogućila je brzo podizanje REST servisa, jasnu strukturu projekta i jednostavno povezivanje s bazom podataka. Integracija Spring Securityja donijela je ugrađenu podršku za različite metode autentifikacije i autorizacije. Komponenta Data JPA koristi se za pojednostavljeni upravljanje relacijskom bazom, dok Starter Web omogućava lako rukovanje HTTP zahtjevima i uspostavljanje API-ja. Spring Boot Starter Test koristi se za različite razine testiranja i osigurava stabilniji kod. Ujedno su korišteni PostgreSQL JDBC Driver za komunikaciju s bazom te jjwt-api, jjwt-impl i jjwt-jackson biblioteke kako bi se omogućila implementacija JWT autentifikacije. Uz to je bilo potrebno uključiti secretsmanager i spring-cloud-aws-starter, što olakšava pohranu osjetljivih podataka i interakciju s AWS servisima. Spring-security-test je poslužio za testne scenarije povezane s provjerama sigurnosnih mehanizama. U build procesu koristi se Spring Boot Maven Plugin za jednostavno upravljanje ovisnosti i kreiranje izvršnih paketa. Na klijentskoj strani korišten je React 18.3.1, koji zahvaljujući modularnom pristupu komponentama i bogatom ekosustavu omogućava razvoj interaktivnih sučelja te brzo uvođenje novih funkcionalnosti.

3. Baza podataka:

Zadaci pohrane i upravljanja podacima riješeni su korištenjem PostgreSQL 15. Implementirana je kroz Aiven, što osigurava stabilan i siguran rad baze, te pojednostavljuje administraciju i nadzor nad podacima. PostgreSQL se pokazuje kao pouzdan izbor zbog podrške za napredne SQL značajke, visokih performansi i pouzdanih mehanizama replikacije.

4. Razvojni alati:

U radu na projektu, odabrani su Visual Studio Code i Eclipse IDE for Java Developers – 2024-03. Visual Studio Code korišten je za frontend razvoj zbog svojih brojnih ekstenzija specifičnih za JavaScript i React okružujući, dok je Eclipse IDE prilagođen radu s Javaom, što dodatno olakšava

JAVAJSKI I REACT FRONTSERV, UOK JE ECLIPSE IDE PRIMJENJUJUĆI RADU S JAVOM, ŠTO OMOGUĆAVA
razvoj i održavanje složenih poslužiteljskih aplikacija. Za upravljanje verzijama koda koristi se Git
2.47, čime je timski rad uvelike pojednostavljen kroz mogućnosti kolaboracije, praćenja povijesti i
grananja.

5. Alati za ispitivanje:

Za potrebe testiranja tijekom razvoja korišten je Postman, koji omogućava brzo i intuitivno slanje
različitih vrsta HTTP zahtjeva te analizu odgovora poslužitelja. Time je ubrzana provjera API
funkcionalnosti te je omogućen jednostavan uvid u integracijske točke između frontend i
backend dijela aplikacije.

6. Alati za razmještaj:

Aplikacija je smještena na Aiven, koji služi kao host server i pojednostavljuje proces puštanja
aplikacije u rad. Zahvaljujući integracijama i kontroli izvedbe, Aiven doprinosi stabilnoj i
skalabilnoj infrastrukturi koja lako odgovara na promjenjive zahtjeve u produkciji.

7. Cloud platforma:

Kao cjelovito rješenje u oblaku koriste se dvije platforme. Frontend je postavljen na Render, koji
pruža pregledne alate za kontinuiranu integraciju i brzi deployment React aplikacija, dok se
pozadinski dio oslanja na AWS. Na taj je način omogućeno iskorištavanje specifičnih prednosti
obje platforme, uključujući globalnu infrastrukturu, sigurnost i dostupnost za backend, te
jednostavan proces razmještaja i održavanja za frontend.

Zaključak:

Ovakav izbor tehnologija nudi stabilnu i prilagodljivu arhitekturu, što olakšava održavanje i
buduće nadogradnje aplikacije. Smatram da precizno određene verzije alata i detaljno planiranje
svakog dijela projekta pomaže u poboljšanju timske suradnje, ubrzavaju izmjene u kodu i
uvođenje novih funkcionalnosti. Kombinacija Spring Boot okvira s Javom, React biblioteke s
JavaScriptom, te pouzdane PostgreSQL baze hostane na Aiven platformi pruža zaista čvrst temelj
za rad. Osim toga, upotreba Rendera i AWS-a za različite aspekte infrastrukture doprinosi visokoj
razini performansi, sigurnosti i dostupnosti. Budući da se tehnologija brzo mijenja, važno je
redovito ažurirati sve korištene komponente i pratiti nove verzije, kako bi se održala stabilnost,
pouzdanost i kvaliteta cijelog sustava. Upravo takav pristup, uz kontinuirano prilagođavanje,
osigurava da će projekt ostati održiv i jednostavan za daljnji razvoj.

+ Add a custom footer

▼ 7. Tehnologije za implementaciju aplikacije

Korištene tehnologije i alati

▶ 8. Upute za puštanje u pogon

▶ 9. Zaključak i budući rad

▶ A. Popis literature

▶ B. Dnevnik promjena dokumentacije

▶ C. Prikaz aktivnosti grupe

6. Ispitivanje programskog rješenja

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>



CognitoAgent /
PlanBot[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

8. Upute za puštanje u pogon

[Edit](#)[New page](#)[Jump to bottom](#)

Toma Begović edited this page 43 minutes ago · [9 revisions](#)

Ovaj odjeljak dokumentacije treba dati detaljne smjernice za instalaciju, konfiguraciju, pokretanje i administraciju aplikacije. Cilj je olakšati postavljanje aplikacije na razvojnem, ispitnom i produkcijskom okruženju.

1. Instalacija

Koraci potrebni za instalaciju svih potrebnih komponenti:

- **Preduvjeti:** Popis potrebnog softvera i njihovih verzija
 - Java verzija 21.0.3 ili viša

- **Preuzimanje:** Upute za preuzimanje izvornog koda

```
git clone https://github.com/CognitoAgent/PlanBot.git
```

```
cd repo
```

Instalacija ovisnosti: Upute za instaliranje ovisnosti.

```
npm install
```

2. Postavke

Detaljne upute za konfiguraciju aplikacije:

- **Konfiguracijske datoteke:** potrebno je konfigurirati application.json datoteku
 - unutar okruženja u kojem se nalazi springboot backend potrebno je dodati sljedeće varijable sa njihovim vrijednostima:
 - AWS_ACCESS_KEY_ID="AKIAT7JJVF2DYS2GQHFJ"
 - AWS_SECRET_ACCESS_KEY="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"
 - KEYSTORE_PASSWORD="stO1rE2"
 - GMAPS=AlzaSyCW9Pfr2aV3vxhxmyVlcW-MFdebbj8wVss
- **Postavke baze podataka:** Upute za inicijalizaciju baze podataka, uključujući migracije i postavljanje inicijalnih podataka.

```
npm run db:migrate
```

```
npm run db:seed
```

3. Pokretanje aplikacije

Upute za pokretanje aplikacije u različitim okruženjima:

- **Razvojno okruženje:**

```
npm run dev
```

- **Produkcijsko okruženje:**

- Prevodenje aplikacije:

```
npm run build
```

- Pokretanje poslužitelja:

```
npm start
```

- **Provjera rada:** Navesti npr. URL (npr. <http://localhost:3000>) .

4. Upute za administratore

Smjernice za administratore aplikacije nakon puštanja u pogon:

- **Pristup administratorskom sučelju:**

- URL za admin panel (npr. /admin).
 - Početni podaci za prijavu (ako postoje).

- **Redovito održavanje:**

- Arhiviranje baze podataka.
 - Pregled logova.
 - Ažuriranje aplikacije (primjer: povlačenje novih verzija iz Git repozitorija i ponovno pokretanje aplikacije).

```
git pull origin main
```

```
npm install
```

```
npm run build
```

```
npm start
```

- **Rješavanje problema:** Kako pristupiti logovima i dijagnosticirati greške (npr. logs/error.log ili docker logs).

5. Primjer za Render platformu (Cloud Deploy)

Render je popularna cloud platforma za jednostavno smještanje aplikacija.

- **Priprema repozitorija:**

- Za postavljanje aplikacije je dovoljno imati git repozitorij koji je moguće povezati s renderom te je moguće iznova ponovno pokretati aplikaciju nakon svake promijene na git repozitoriju

- **Postavljanje na Render:**

- Prijavite se na [Render](#).
- Kreirajte novi **Static Site** i povežite ga s vašim GitHub repozitorijem.
- Unesite ime aplikacije
- Unesite URL vašeg repozitorija na gitu
- Unesite naziv željene grane
- Unesite e-mail adresu za vaš korisnički račun na gitu
- Odaberite direktorij u repozitoriju u kojem se nalazi aplikacija koju puštate u pogon
- Odaberite naredbu koja se automatski izvršava prije svakog puštanja u pogon (npm build)
- Dodajte environment varijable (npr. DATABASE_URL, API_KEY).
- Moguće je konfigurirati još neke postavke, kao unošenje vlastitog naziva za domenu, ali nije potrebno da bi aplikacija bila uspješno puštena u pogon

- **Pokretanje aplikacije:**

Render će automatski preuzeti repozitorij, instalirati ovisnosti i pokrenuti aplikaciju. Nakon deploja, aplikaciji možete pristupiti putem generiranog URL-a (npr. <https://my-web-app.onrender.com>).

Opis prisutpa aplikaciji na javnom poslužitelju

Pristup aplikaciji

Kako biste pristupili aplikaciji, potrebno je otvoriti sljedeći link: <https://my-web-app.onrender.com>

planbot-9s64.onrender.com Link će vas preusmjeriti na stranicu za prijavu ("Sign in"), gdje možete odabrati prijavu koristeći postojeći korisnički račun, prijavu putem Google računa ili registraciju.

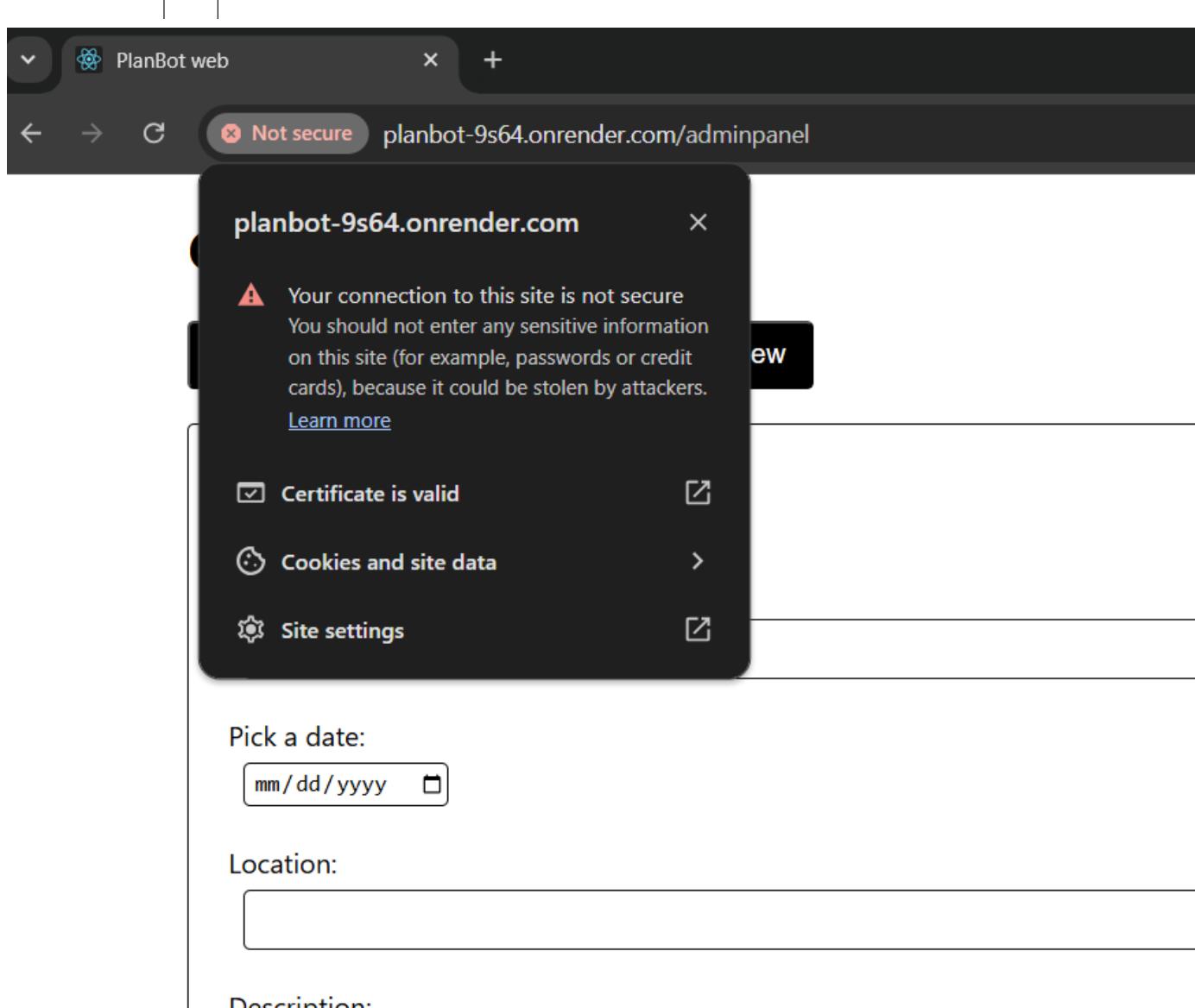
Ukoliko odaberete:

- prijavu s postojećim računom: možete se prijaviti postojećim računom:
 - username: ana
 - password: anapassword
 - Napomena: Korisnik "ana" je administrator i ima mogućnost pristupa stranici "Admin View"
- prijava putem Google računa: Slijedite upute koje će se prikazati u novom prozoru
- registracija novog korisnika:
 - username: Možete unijeti bilo koje ime, pod uvjetom da ime nije već zauzeto
 - Email address: Mora biti valjana (npr. mora sadržavati znak "@" i točku)
 - Password: Može biti bilo koji niz znakova
 - Confirm password: Lozinka mora biti identična prethodno unesenoj

VAZNO: Ako registracija ili prijava nisu uspješne, pročitajte sljedeće: Kako bismo omogućili rad aplikacije putem HTTPS protokola, koristimo certifikat izrađen pomoću AWS Certificate Manager-a (<https://aws.amazon.com/certificate-manager/>). Ovisno o uređaju i pregledniku koji koristite, certifikat može biti prepoznat kao važeći ili nevažeći.

Primjeri uspješne validacije certifikata:

The screenshot shows a web browser window with the URL <https://planbot-9s64.onrender.com/adminpanel>. A context menu is open over the address bar, displaying "Site information for planbot-9s64.onrender.com". The menu includes "Connection secure" (with a lock icon) and "Clear cookies and site data...". Below the browser window, a modal dialog titled "Create New Event" is displayed. The dialog has fields for "Event Title" (with a text input field), "Pick a date" (with a date input field), and "Location" (with a text input field).



Ako se suočavate s poteškoćama prilikom registracije ili prijave, postoji alternativan način pristupa aplikaciji kojim se onemogućava provjera certifikata. Certifikat se u ovom slučaju ignorira. Slijedite upute ovisno o vašem operativnom sustavu:

- Za Windows korisnike: -ako trenutno imate otvoren Chrome prozor, zatvorite ga! Navedena naredba će se uspjesno pokrenuti samo ako nije otvoren ni jedan Chrome prozor -unutar Command Prompt-a pokrenite sljedeću naredbu:
 - "C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" --ignore-certificate-errors
 - Napomena: pretpostavlja se da je Chrome instaliran na navedenoj putanji
- Za macOS korisnike:
 - unutar Terminala pokrenite sljedeću naredbu:
 - open -a "Google Chrome" --args --ignore-certificate-errors

Open a Google Chrome args ignore certificate errors

- Za Linux korisnike:

- unutar Terminala pokrenite sljedeću naredbu:
- google-chrome --ignore-certificate-errors
- ako Chrome aplikacija nije dio default path-a, potrebno je specificirati punu putanju, npr
- /usr/bin/google-chrome --ignore-certificate-errors

- Nakon pokretanja aplikacije na prijašnje navedeni način, možete pristupiti stranici putem sljedećeg linka: <https://planbot-9s64.onrender.com>

+ Add a custom footer

▼ Pages 13

Find a page...

▶ [Home](#)

▶ [1. Opis projektnog zadatka](#)

▶ [2. Analiza zahtjeva](#)

▶ [3. Specifikacija zahtjeva sustava](#)

▶ [4. Arhitektura i dizajn sustava](#)

▶ [5. Arhitektura komponenata i razmještaja](#)

▶ [6. Ispitivanje programskog rješenja](#)

▶ [7. Tehnologije za implementaciju aplikacije](#)▼ [8. Upute za puštanje u pogon](#)

Opis prisutpa aplikaciji na javnom poslužitelju

▶ [9. Zaključak i budući rad](#)▶ [A. Popis literature](#)▶ [B. Dnevnik promjena dokumentacije](#)▶ [C. Prikaz aktivnosti grupe](#)

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>



[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

9. Zaključak i budući rad

[Edit](#)[New page](#)[Jump to bottom](#)

Lea Gojić edited this page 1 hour ago · [7 revisions](#)

dio 2. revizije

U ovom poglavlju potrebno je napisati osvrt na vrijeme izrade projektnog zadatka, koji su tehnički izazovi prepoznati, jesu li riješeni ili kako bi mogli biti riješeni, koja su znanja stečena pri izradi projekta, koja bi znanja bila posebno potrebna za brže i kvalitetnije ostvarenje projekta i koje bi bile perspektive za nastavak rada u projektnoj grupi. Potrebno je točno popisati funkcionalnosti koje nisu implementirane u ostvarenoj aplikaciji.

Usvojena znanja

- Stečeno je značajno iskustvo u razvoju aplikacija koristeći Spring Boot okvir. Posebno su usvojena sljedeća znanja:
 - Razumijevanje pojmove poput bean, annotacija i transakcija, kao i njihove primjene u kontekstu aplikacije
 - Korištenje HTTP metoda poput POST, GET i DELETE unutar kontrolera za rukovanje zahtjevima
 - Rad s JpaRepository za manipulaciju podacima u bazi podataka te implementacija relacija između entiteta (primjerice između korisnika i objava)
 - Razumijevanje sigurnosnih konfiguracija, uključujući autentifikaciju i autorizaciju pomoću JWT tokena
 - Korištenje Google API keya za prikaz Google Karte u aplikaciji
 - Detaljno razumijevanje strukture i toka aplikacije: Kroz pisanje testova u Seleniumu i analiziranje elemenata, usvojena su znanja o tome kako aplikacija funkcionira na razini klijenta te kako se pojedine komponente povezuju u cjelinu.
 - Primjena automatiziranog testiranja i asertivnih provjera: Korištenjem PyTest-a i Selenium WebDriver-a, usvojene su tehnike izrade automatiziranih testova, metode za pronalaženje i manipuliranje HTML elementima, te način na koji se kreiraju i provjeravaju uvjeti (asercije) za različite scenarije – od regularnih do rubnih i negativnih testova.
 - Nauči se modelirati arhitekturu sustava, razumjeti odnose među komponentama te njihovu implementaciju i raspodjelu u stvarnom okruženju.

Tehnički izazovi

- znacajniji izazov bio je implementacija HTTPS certifikata kako bi aplikacija bila dostupna putem sigurnog protokola.
 - Certifikat je izrađen pomoću AWS Certificate Manager-a
 - Prepoznato je da neki preglednici (ovisno o platformi) certifikat ne registriraju kao valjan
 - Potencijalno rješenje: ponovno izdavanje certifikata koristeći drugu platformu ili ručna instalacija certifikata na uređajima korisnika.
- integracija različitih komponenti projekta (backend i frontend) te osiguranje njihove

- integracija različitih komponenta projekta (backend i frontend) te osiguranje njihove funkcionalnosti u razvojnim i producijskim okruženjima
 - dohvaćanje podataka i njihovo prikazivanje svaki put nakon promjene svojstva nekog podatka
 - riješeno je korištenjem efekta za dohvaćanje podataka i korištenjem stanja za spremanje vrijednosti o podatcima te postavljanjem ovisnosti efekta o stanju

Funkcionalnosti koje nisu implementirane

- Notifikacijski sustav za obavještavanje korisnika o novim događajima ili promjenama u objavama
- Integracija Google kalendara, što bi omogućilo sinkronizaciju događaja iz aplikacije s osobnim kalendarima korisnika
- Označivanje nekog događaja finalnim
- Mogućnost označivanja dogadaja kao javnim ili privatnim
- Opcija pretraživanja
- Mogućnost mijenjanja postavki

+ Add a custom footer

▼ Pages 13

▶ [Home](#)

▶ [1. Opis projektnog zadatka](#)

▶ [2. Analiza zahtjeva](#)

▶ [3. Specifikacija zahtjeva sustava](#)

▶ [4. Arhitektura i dizajn sustava](#)

▶ [5. Arhitektura komponenata i razmještaja](#)

▶ [6. Ispitivanje programskog rješenja](#)



▶ [7. Tehnologije za implementaciju aplikacije](#)

▶ [8. Upute za puštanje u pogon](#)

[9. Zaključak i budući rad](#)

▶ [A. Popis literature](#)

▶ [B. Dnevnik promjena dokumentacije](#)

▶ [C. Prikaz aktivnosti grupe](#)

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>



[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

A. Popis literature

[Edit](#)[New page](#)[Jump to bottom](#)

JosipGalic edited this page 3 hours ago · [4 revisions](#)

Kontinuirano osvježavanje Popisati sve reference i literaturu koja je pomogla pri ostvarivanju projekta.

1. Programsko inženjerstvo, FER ZEMRIS, <http://www.fer.hr/predmet/proinz>
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book", Department of Electrical and Computer Engineering, Rutgers University, [<http://www.ece.rutgers.edu/~marsic/> (<http://www.ece.rutgers.edu/%CB%9Cmarsic/>) books/SE
5. The Unified Modeling Language, <https://www.uml-diagrams.org/>
6. Astah Community, <http://astah.net/editions/uml-new>
7. Google Maps Platform, <https://developers.google.com/maps/documentation/embed/embedding-map>
8. SpringBoot official website, <https://spring.io/projects/spring-boot>
9. JSON Web Tokens, <https://jwt.io/>
10. Maven Repository, <https://mvnrepository.com/>
11. Java SE 21 & JDK 21, <https://docs.oracle.com/en/java/javase/21/docs/api/index.html>
12. React, <https://react.dev/>

[+ Add a custom footer](#)

CognitoAgent /
PlanBot[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

B. Dnevnik promjena dokumentacije

[Edit](#)[New page](#)[Jump to bottom](#)

Toma Begović edited this page 42 minutes ago · [20 revisions](#)

Rev.	Opis promjene/dodataka	Autori	Datum
0.1	Opis projektnog zadatka(1. stranica)	Lea Gojić	31.10.2024.
0.2	Dodani funkcionalni zahtjevi	Lea Gojić	3.11.2024.
0.3	Dodan opis baze podataka i njene tablice	Josip Galić	12.11.2024.
0.4	Dodan stil arhitekture	Nina Karavanić	12.11.2024.
0.5	Analiza zahtjeva	Lea Gojić	13.11.2024.
0.6	Uređen opis projektnog zadatka	Lea Gojić	15.11.2024.
0.7	Dodani sekundarni aktori	Lea Gojić	15.11.2024.
0.8	Dodani sastanci u aktivnostima grupe	Josip Galić	15.11.2024.
0.9	U arhitekturi i dizajnu sustava dodan dijagram baze podataka	Nina Karavanić	15.12.2024.
1.0	Dodani obrasci uporabe, dijagrami obrazaca uporabe, sekvencijski dijagrami	Ema Mamić	16.12.2024
1.1	Korištene tehnologije i alati	Lea Gojić	5.1.2025.
1.2	Dijagram razmještaja	Borna Maričak	20.1.2025.
1.3	Analiza zahtjeva, Arhitektura i dizajn, dorada	Nina Karavanić	21.1.2025.
1.4	Azuriranje opisa projektnog zadatka	Josip Galić	22.1.2025.
1.5	Azuriranje opisa obrazaca uporabe	Josip Galić	22.1.2025.
1.6	Opis prisutpa aplikaciji na javnom poslužitelju	Nina Karavanić	23.1.2025.
1.7	Zaključak i buduci rad	Nina Karavanić	24.1.2025.
1.8	Zaključak i budući rad	Toma Begović	24.1.2025.
1.9	Popis literature, dodani linkovi	Nina Karavanić	24.1.2025.

2.0	Azuriranje prikaza aktivnosti grupe	Josip Galić	24.1.2025.
2.1	Dovrsena home stranica	Nina Karavanić	24.1.2025.
2.2	Dijagram komponenti	Lea Gojić	24.1.2025.
2.3	Ispitivanje komponenti i sustava	Borna Maričak	24.1.2025.
2.4	Ispravljanje gramatičkih pogrešaka prisutnog dijela dokumentacije	Nina Karavanić	24.1.2025.
2.5	Završna prezentacija	Lea Gojić	24.1.2025.
2.6	Ključni izazovi i rješenja	Nina Karavanić	24.1.2025.
2.7	Upute za puštanje u pogon	Toma Begović	24.1.2025.

Evidencija promjena sadrži popis promjena izvršenih tijekom cijelo životnog trajanja predmeta evidencije (dокументacije, projekta i sl.). Osnova svrha je praćenja napretka svake promjene na temelju njezina preispitivanja, odobrenja (ili odbijanja), provedbe, kao i zaključenja. Štoviše, dobar dnevnik promjena sadrži i datum promjene i njegov utjecaj na projekt u smislu rizika, vremena i troškova. Sve te promjene prenose se dionicima. Štoviše, odbačene promjene također su uključene u povijest promjena.

Zašto? Olakšano praćenje ključnim dionicima.

+ Add a custom footer

▼ Pages 13

Find a page...

- ▶ [Home](#)
- ▶ [1. Opis projektnog zadatka](#)
- ▶ [2. Analiza zahtjeva](#)
- ▶ [3. Specifikacija zahtjeva sustava](#)

- ▶ [7. Tehnologije za implementaciju aplikacije](#)
- ▶ [8. Upute za puštanje u pogon](#)
- ▶ [9. Zaključak i budući rad](#)
- ▶ [A. Popis literature](#)
- B. Dnevnik promjena dokumentacije**
- ▶ [C. Prikaz aktivnosti grupe](#)

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>



[Code](#)[Issues 1](#)[Pull requests](#)[Actions](#)[Projects](#)[Wiki](#)[Security](#)

C. Prikaz aktivnosti grupe

[Edit](#)[New page](#)[Jump to bottom](#)

Nina Karavanić edited this page 9 minutes ago · [8 revisions](#)

#Dnevnik sastajanja

1. sastanak

- Datum: 9. listopada 2024.
- Prisustvovali: N.Karavanić, T.Begović, E.Mamić, L.Gojšić, B.Maričak, J.Galić
- Teme sastanka:
 - Upoznavanje i razgovor o temi
 - Dogovor oko teme je bio uspješan

2. sastanak

- Datum: 15. listopada 2024.
- Prisustvovali: N.Karavanić, T.Begović, E.Mamić, L.Gojšić, B.Maričak, J.Galić
- Teme sastanka:
 - Odabir tehnologija koje ćemo koristiti u projektu
 - Dogovor oko tehnologija je bio uspješan
 - Podjela uloga za projekt
 - Podijelili smo uloge za projekt

3. sastanak

- Datum: 18. listopada 2024.
- Prisustvovali: N.Karavanić, J.Galić
- Teme sastanka:
 - Instaliranje potrebnih alata i pokretanje SpringBoot Java projekta

4. sastanak

- Datum: 21. listopada 2024.
- Prisustvovali: N.Karavanić, J.Galić
- Teme sastanka:
 - Nadograđivanje back-end koda
 - Kreiranje React stranice

4. sastanak

4. sastanak

- Datum: 28. listopada 2024.
- Prisustvovali: N.Karavanić, J.Galić
- Teme sastanka:
 - Početak rada na security-u u SpringBoot Javi

5. sastanak

- Datum: 7. studenoga 2024.
- Prisustvovali: N.Karavanić, T.Begović, E.Mamić, L.Gojšić, B.Maričak
- Teme sastanka:
 - Dogovor o nastavku rada na projektu i dokumentaciji

6. sastanak

- Datum: 13. studenoga 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:
 - Početak rada na deploy-u aplikacije

7. sastanak

- Datum: 14. studenoga 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:
 - Nastavak rada na deploy-u aplikacije

8. sastanak

- Datum: 15. studenoga 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:
 - Nastavak rada na deploy-u aplikacije

9. sastanak

- Datum: 10. prosinca 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić

- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:

- Kreiranje objava

10. sastanak

- Datum: 15. prosinca 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:

- Mogućnost filtracije objava

11. sastanak

- Datum: 19. prosinca 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:

- Stvaranje stranice za vlastite objave

12. sastanak

- Datum: 3. siječnja 2024.
 - Prisustvovali: N.Karavanić, T.Begović, J.Galić
 - Teme sastanka:
- Mogućnost prihvatanja objave klikom na Accept

13. sastanak

- Datum: 6. siječnja 2024.
 - Prisustvovali: N.Karavanić, T.Begović, J.Galić
 - Teme sastanka:
- Početak embedda Google Maps u stranicu, dodavanje prijedloga

14. sastanak

- Datum: 12. siječnja 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:

- Dodavanje komentara na objavu

15. sastanak

- Datum: 15. siječnja 2024.
- Prisustvovali: N.Karavanić, T.Begović, L.Gojić, E.Mamić, B.Maričak, J.Galić
- Teme sastanka:
 - Raspodjela zadataka za testiranje i wiki, te nastavak rada na funkcionalnostima aplikacije

16. sastanak

- Datum: 16. siječnja 2024.
- Prisustvovali: N.Karavanić, T.Begović, J.Galić
- Teme sastanka:
 - Završen embedd Google Karte u aplikaciju

12. sastanak

- Datum: 21. siječnja 2024.
- Prisustvovali: N.Karavanić, T.Begović, L.Gojić
- Teme sastanka:
 - Uređivanje aplikacije, dodavanje CSS-a

Plan rada

Tablični/Gantt/Kanban prikaz

- Prikaz vremenskog plana rada ključnih aktivnosti (tjedna granulacija)
- Uključuje akronime angažiranih članova tima TODO: primjer

Tablica aktivnosti

Kontinuirano osvježavanje

Napomena: Doprinose u aktivnostima treba navesti u satima po članovima grupe po aktivnosti. Potrebno je navesti koliko je sati koja osoba uložila u pojedinu komponentu, možete oblikovati tablicu ili ispisati za svaku osobu.

- Upravljanje projektom: Nina Karavanić
- Opis projektnog zadatka: Nina Karavanić(1h), Josip Galić(1h)
- Funkcionalni zahtjevi: Lea Gojšić(5h), Nina Karavanić(3h), Josip Galić(2h)
- Opis pojedinih obrazaca: B.Maričak(4h)
- Dijagram obrazaca: E.Mamić(4h)
- Sekvencijski dijagrami: E.Mamić(4h)
- Opis ostalih zahtjeva : Lea Gojšić(3h)
- Arhitektura i dizajn sustava: Nina Karavanić(2h)
- Baza podataka i tablice: Josip Galić (2h), B.Maričak(2h)
- Dijagram razreda: Ema Mamić (2h)
- Dijagram stanja: Ema Mamić (3h)
- Dijagram aktivnosti: Ema Mamić (3h)
- Dijagram komponenti: Ema Mamić (3h)
- Korištene tehnologije i alati: Nina Karavanić (2h), Lea Gojšić (2h), Josip Galić (2h)
- Ispitivanje programskog rješenja: Lea Gojšić (6h), Borna Maričak(6h)
- Dijagram razmještaja: Ema Mamić (3h), Nina Karavanić (1h)
- Upute za puštanje u pogon: Nina Karavanić (2h)
- Dnevnik sastajanja: Josip Galić(1h)
- Zaključak i budući rad: Nina Karavanić (2h), Lea Gojšić (1h), Borna Maričak(1h), Josip Galić (1h), Toma Begović (1h)
- Popis literature: Nina Karavanić (1h)
- SpringBoot Java: Nina Karavanić(50h), Josip Galić(15h)
- nadograđivanje aplikacije: Nina Karavanić(10h), Josip Galić(8h), Toma Begović (8h)
- ReactJS: Toma Begović(50h), Josip Galić(30h)
- izrada početne stranice: Josip Galić (2h)
- izrada baze podataka: Nina Karavanić (3h)
- deploy: Nina Karavanić(9h), Josip Galić(9h), Toma Begović (9h)
- spajanje s bazom podataka: Nina Karavanić (1h)
- izrada prezentacije: Nina Karavanić(2h), Lea Gojšić (5h)

Dijagram pregleda promjena

Prenijeti dijagram pregleda promjena nad datotekama projekta. Potrebno je na kraju projekta

generirane grafove s githuba prenijeti u ovo poglavlje dokumentacije. Dijagrami za vlastiti projekt se mogu preuzeti s github.com stranice, u izborniku Repository, pritiskom na stavku Contributors.

Kjučni izazovi i rješenja

- Zaključno Opis izazova: Tijekom provedbe projekta, naš tim se suočio s nekoliko značajnih izazova:
- Zdravstveni problemi člana tima: Jedan od članova tima imao je zdravstvene probleme, što je zahtijevalo prilagodbu plana rada kako bi mu omogućili da svoj dio posla obavi s odgodom.
- Problemi s CORS (Cross-Origin Resource Sharing): Susreli smo se s tehničkim problemima povezanim s CORS politikama, što je otežavalo integraciju frontend i backend dijelova naše aplikacije.
- Poteškoće korištenja repozitorija: Jedan član tima, zadužen za rad na web stranici, nije na vrijeme otvorio repozitorij, što je rezultiralo poteškoćama u razvoju. Ova situacija bila je važna lekcija o važnosti pravovremenog pristupa i korištenja repozitorija.
- Nedostatak jednog člana tima: Planirani broj članova tima bio je sedam, no zbog okolnosti smo radili u sastavu od šest osoba, što je povećalo opterećenje na ostale članove tima.

Rješenja i naučene lekcije:

- Fleksibilnost i podrška: Uvedene su prilagodbe u plan projektiranja kako bismo omogućili fleksibilniji raspored. Ovo je uključivalo fleksibilne rokove i redovite provjere stanja zdravlja i dobrobiti članova tima.
- Tehničko usavršavanje: Za rješavanje problema s CORS, ostvario se bolji uvid u problem proučavanjem literature i web stranica koje govore o navedenom.
- Promoviranje odgovornosti: Incident s repozitorijem poslužio je kao temelj za razvoj boljih praksi u upravljanju i korištenju repozitorija, osiguravajući da svi članovi tima imaju adekvatan pristup i obuku za rad s alatima koji su im na raspolaganju.

+ Add a custom footer

▼ Pages 13

Find a page...

▶ [8. Upute za puštanje u pogon](#)▶ [9. Zaključak i budući rad](#)▶ [A. Popis literature](#)▶ [B. Dnevnik promjena dokumentacije](#)▼ [C. Prikaz aktivnosti grupe](#)

Plan rada

Tablica aktivnosti

Dijagram pregleda promjena

Kjučni izazovi i rješenja

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/CognitoAgent/PlanBot.wiki.git>

